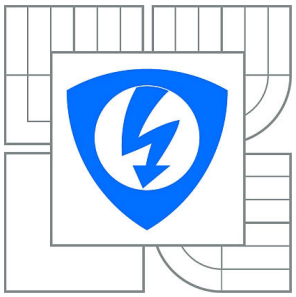
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

LOKALIZACE GEOGRAFICKÉ POZICE ÚTOČNÍKA V INTERNETU

LOCALIZATION OF GEOGRAPHIC POSITION ATTACKER IN THE INTERNET

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

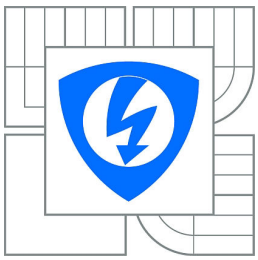
AUTOR PRÁCE — Bc. FRANTIŠEK KUBALÍK

AUTHOR

VEDOUCÍ PRÁCE — Ing. LUKÁŠ VERNER

SUPERVISOR

BRNO 2014



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bc. František Kubalík	***ID:***	72950
Ročník:	2	***Akademický rok:***	2013/2014

NÁZEV TÉMATU:

Lokalizace geografické pozice útočníka v internetu

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s principy vyhodnocování fyzické polohy stanic v síti Internet se zaměřením na aktivní metody IP geolokace. Dále se seznamte s experimentální sítí PlanetLab – http://www.planet-lab.org/. Nastudujte a naprogramujte lokalizační algoritmus Octant. Vytvořte modifikaci algoritmu se zaměřením na eliminaci lokalizace stanice v místech s malou pravděpodobností výskytu. Analyzujte přesnost navržených algoritmů pomocí uměle vytvořených vstupních dat a porovnejte jejich přesnost dostupnými metodami IP geolokace.

DOPORUČENÁ LITERATURA:

[1] MATRAY, P., HAGA, P., LAKI, S., CSABAI, I., VATTAY, G. On the network geography of the Internet. International Conference on Computer Communications. IEEE, 2011.

[2] WONG, B., STOYANOV, I., SIRER, E. Octant: A comprehensive Framework for the Geolocalization of Internet Hosts. Symposium on Networked System Design and Implementation. USENIX, 2007.

[3] PlanetLab Consortium. PlanetLab: An open platform for developing, deploying, and accessing planetary-scale services. URL: <http://www.planet-lab.org> [cit. 08. 10. 2013].

Termín zadání:	10.2.2014	***Termín odevzdání:***	30.5.2014

Vedoucí práce: Ing. Lukáš Verner
Konzultanti diplomové práce:

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

Abstrakt

Tato diplomová práce se zabývá vyhodnocením geografické polohy stanic v síti Internet pomocí zpoždění a implementováním metody Octant. První část práce se věnuje seznámení s principy vyhodnocování fyzické polohy stanic v síti Internet se zaměřením na aktivní metody IP geolokace. Větší prostor je zde věnován geolokačnímu algoritmu Octant a podrobnějšímu vysvětlení jeho principů. Na konci první části je také seznámení s experimentální sítí PlanetLab. Ve druhé části je popsán návrh a realizace programu v programovacím jazyce C#, který lokalizuje hledanou stanici zmíněným lokalizačním algoritmem Octant. Popis realizace je doprovázen vysvětlenými ukázkami použitých tříd a jejich podrobným vysvětlením. Jsou zde také uvedeny výsledky metody na vybrané IP adresy stanic se známou polohou pro zjištění chyb neboli odchylek metody od skutečné polohy hledané stanice. Na konci této části je uvedeno porovnání s ostatními aktivními ale i pasivními geolokačními metodami.

Klíčová slova

Octant, PlanetLab, geolokace, lokalizace, zpoždění, C#

Abstract

This thesis deals with the evaluation of the geographical location of the Internet using delay and implementating method Octant. The first part is devoted to familiarization with the principles of evaluation of the physical position of the stations on the Internet, with a focus on active methods for IP Geolocation. Here it is more space devoted to geolocation Octant algorithm and a more detailed explanation of its principles. At the end of the first part is also familiar with the experimental network PlanetLab. The second part describes the design and implementation of the program in programming language C#, which locates the searched local station with mentioned localization algorithm Octant. Description realization is accompanied by explained examples of the classes with a detailed explanation. Here are also the results of the methods on selected IP addresses of stations with known locations to detect errors or deviations of method from the actual position of searched station. At the end of this section is a comparison with other active as well as passive geolocation methods.

Keywords

Octant, PlanetLab, geolocation, localization, delay, C#

KUBALÍK, F. *Lokalizace geografické pozice útočníka v internetu.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2014. 51 s. Vedoucí diplomové práce Ing. Lukáš Verner.

Prohlášení

Prohlašuji, že svou diplomovou práci na téma „Lokalizace geografické pozice útočníka v internetu“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne ……. .....

podpis autora

Poděkování

Děkuji vedoucímu práce Ing. Lukáši Vernerovi za velmi užitečnou metodickou pomoc a cenné rady při zpracování diplomové práce.

V Brně dne ……. .....

podpis autora

SIX research centre
sensor, information and communication systems

Faculty of Electrical Engineering
and Communication

Brno University of Technology
Technicka 12, CZ-61600 Brno, Czechia

http://www.six.feec.vutbr.cz

Výzkum popsaný v této diplomové práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

2007–13
OP Výzkum a vývoj pro inovace

Obsah

Úvod

Cílem této diplomové práce je seznámit se s principy vyhodnocování fyzické polohy stanic v síti Internet se zaměřením na aktivní metody IP geolokace a nastudovat a naprogramovat lokalizační algoritmus Octant ve zvoleném programovacím jazyce. Dalším cílem je analyzovat přesnost navrženého algoritmu a porovnat jeho přesnost s dostupnými metodami geolokace.

Na následujících stránkách jsou tyto úkoly podrobněji popsány. V první hlavní kapitole se věnuji představení základního pojmu geolokace a rozdělení metod geolokace na aktivní a pasivní. Skupina pasivních metod vyhledává polohu cílové stanice například v databázi podle IP adresy a patří do ní třeba metoda GeoIP. Druhá skupina se podílí aktivně na měření v síti s následnou lokalizací cílového uzlu. Do této skupiny patří například metody CBG (Constraint Based Geolocation), SOI (Speed of Internet) a hlavně metoda Octant. Metodě Octant je věnována podstatná část teroetické části a jsou zde podrobněji popsány principy a mechanismy, se kterými metoda pracuje - od vypočítávání hranic omezení až po výpočet pravděpodobné polohy cílové stanice. Dále je v teoretické části představena experimentální síť PlanetLab a poskytnuty jen nejzákladnější informace o programovacím jazyku C#.

V praktické části práce je podrobně popsána struktura programu vyvinutém v jazyce C# realizující metodu Octant. Dále je v této části popsáno její testování na reálných datech v několika fázích a ukázka lokalizace vybrané cílové stanice, která je ilustrována grafickými znázorněními. V poslední části praktické části je pak porovnání výsledků lokalizovaných stanic s ostatními metodami geolokace.

1 GEOLOKACE CÍLOVÉ STANICE

Obecně geolokace neboli geografická lokalizace znamená zjištění geografických údajů fyzické polohy nějakého objektu, který hledáme. IP (Internet Protocol) geolokace je tedy metoda, která se používá pro zjištění geografických údajů fyzické polohy nějakého síťového zařízení, které je připojené k internetu (počítač, mobilní telefon, server apod.). Zjištění polohy těchto zařízení se dá využít v mnoha službách a aplikacích v internetovém světě. Jsou jimi například:

- **Online reklama**

  Efektivní šíření reklamy na správnou cílovou skupinu. Geolokací je zjištěna geografická oblast cílové skupiny a díky tomu jsou jí nabízeny jen služby a produkty v jejím okolí.
- **Aktuálnost z hlediska polohy**

  Zajištění aktuálních informací (počasí, zprávy, kulturní akce) z okolí uživatelů internetu, kteří často cestují. Je požadováno, aby byly cílené automaticky.
- **Vyhledávání dopravního spojení**

  Služba podobná jako v předchozím případě. Zde se geolokace využívá k určení aktuální polohy pohybujícího se uživatele a následně je mu nabídnuto dopravní spojení v jeho okolí.
- **VoIP telefonie**

  Protože se s rozvojem internetové sítě rozvíjí i odvětví VoIP (Voice over Internet Protocol), je i zde přání uživatelů zjistit místo, kde se nachází protějšek této komunikace. Toho lze využít i při krizové situaci, kdy je volající ohrožen na životě nebo zdraví a není schopen komunikovat.
- **Ochrana proti podvodům**

  Zde slouží geolokace pro odhalování podvodů s kreditními kartami. Zákazník si zvolí oblast, kde kreditní kartu obvykle používá a tím může být odhaleno zneužití v případě použití kreditní karty v jiné oblasti.

- **Boj proti spamu**

  V tomto případě je boj proti spamu založen na myšlence, že spamy jsou odesílány spíše ze zemí, které nemají vyvinutou antispamovou legislativu. Pomocí geolokace jsou tedy detekovány datové přenosy z těchto zemí a přenesená data jsou pak označena jako podezřelá, nebo jsou rovnou vymazána. Je zde však riziko, že jsou jako spam identifikovány i běžné datové přenosy.

- **Zábava**

  Díky geolokace vznikají i sociální sítě založené na jejím principu. Jsou to například Foursquare a Gowalla. U Foursquare chodí uživatelé do různých budov a schází se tam s uživateli, kteří jsou v dané budově také. Gowalla je v principu hra, kde jsou uživatelé pomocí geolokace naváděni na daná místa, kde se nacházejí předměty. Tyto předměty uživatelé přesouvají na jiná místa. Na internetových stránkách je pak možné sledovat pohyb těchto předmětů po světě [1].

V současné době pro geografickou lokalizaci v IP sítích existují dva typy metod, které se liší použitými principy a přesností - pasivní a aktivní metody. Pasivní metody využívají statických záznamů především v databázích, případně prohledávání DNS (Domain Name System) záznamů. Aktivní metody využívají korelace mezi síťovým zpožděním a geografickou vzdáleností [2], [3].

1.1 Pasivní geolokační metody

Pasivní geolokační metody jsou v současnosti častěji používanými ke geolokaci. Není nutno se zařízením, jehož poloha je zjišťována, nijak komunikovat [2]. Jak již bylo naznačeno výše, využívají se statické záznamy v databázích, prohledávání DNS nebo geolokace pomocí dodatečných zdrojů informací jako Wifi sítí nebo GSM (Global System for Mobile). Výhodou pasivních technik je jejich rychlost a relativní přesnost [3].

1.1.1 Použití databázového systému

Geolokační informace se získávají prostřednictvím porovnávání záznamů IP adres s databází organizace pro přidělování IP adres IANA (Internet Assigned Numbers Authority). Ta se pak dělí ještě na několik koordinačních středisek podle lokality, kterou databáze spravují. Nejznámější databázové geolokační služby jsou Whois, GeoIP, IP2Location, Quova, Geobytes

a HostIP.info. Protože jsou tyto databáze velice náročné na správu, je většina kvalitních databází komerčních. Kvůli manuální správě se v databázích často vyskytují chybné záznamy - například stejná pozice pro skupinu IP adres, které patří zařízením nacházejícím se na různých místech. Dalším problémem je konvergence při přesunu stanice do nové lokace. Při nástupu IPv6 navíc dojde k výraznému zvětšení objemu těchto databází [3]. Níže jsou popsány první dvě zmiňované služby.

Whois

Veřejná databáze Whois je současnou nejpoužívanější databází. Obsahuje záznamy zaměřené na evidenci údajů o majitelích internetových domén a IP adres, což jsou například jméno správce domény, kontaktní adresa, email, popřípadě telefonní číslo. K zjišťování informací o doméně slouží programový nástroj Whois, kde se jako parametr udává hledaná doména. Odpověď databáze je ve formátu textového souboru. Velkou nevýhodou této odpovědi je, že neobsahuje pokaždé informace ve stejném textovém formátu a proto je velmi těžké pro programátora vytvořit funkci, která by dokázala zpracovat všechny formáty záznamů v databázi Whois [1]. Na Obr. 1.1 je uvedena část výstupu programu Whois pro hledanou doménu vutbr.cz z internetových stránek www.whois-search.com [4].

```
domain:      vutbr.cz
registrant:  SB:VUTBR-CZ
admin-c:     CID:VSLAMA
admin-c:     CID:IHAZMUK
nsset:       NSS:VUTBR:1
keyset:      KEYSET-VUTBR.CZ:1
registrar:   REG-GENREG
registered:  19.05.1994 02:00:00
changed:     04.04.2012 00:54:29
expire:      12.10.2014

contact:     SB:VUTBR-CZ
org:         Vysoke uceni technicke v Brne
name:        Vysoke uceni technicke v Brne
address:     Antoninska 548/1
address:     Brno
address:     601 90
address:     CZ
e-mail:      vit.slama@vutbr.cz
registrar:   REG-GENREG
created:     10.08.2001 22:13:00
changed:     10.02.2011 13:30:09

contact:     CID:VSLAMA
org:         Vysoké učení technické v Brně
name:        Vít Sláma
address:     Antonínská 548/1
address:     Brno
address:     60190
address:     CZ
registrar:   REG-GENREG
created:     15.02.2007 10:25:00

contact:     CID:IHAZMUK
org:         Vysoké učení technické v Brně
name:        Ivo Hažmuk
address:     Antonínská 548/1
address:     Brno
address:     601 90
address:     Jihomoravský kraj
address:     CZ
registrar:   REG-GENREG
created:     06.10.2008 17:30:01
```

Obr. 1.1 Výstup programu Whois

GeoIP

GeoIP obsahuje knihovnu jazyka C, která umožňuje najít zemi, ze které pochází daná IP adresa nebo název počítače. K tomu je využita souborová databáze, která obsahuje bloky IP jako klíče a země, města, PSČ a zeměpisné souřadnice jsou pak hodnotami těchto klíčů. Pro některé IP adresy stanic obsahuje databáze GeoIP přesné informace o poloze včetně města a geografických souřadnic. Pro jiné IP adresy ale většinou dokáže určit pouze zemi původu. V takovém případě jsou zeměpisné souřadnice hledané stanice nastaveny na střed země jejich původu [1].

Na Obr. 1.2 je v levé části uvedena opět část výstupu pro hledanou doménu vutbr.cz, která obsahuje město a jeho zeměpisné souřadnice. V pravé části obrázku je zobrazena část výstupu pro doménu centrum.cz, která obsahuje pouze zemi původu a zeměpisné souřadnice jejího středu. Vyhledání domén bylo provedeno na stránkách www.geoiptool.com [5].

Obr. 1.2 Výstup programu GeoIP

1.1.2 Použití prohledávání DNS

V 90. letech 20. století se objevila myšlenka vložit údaje o poloze zařízení přímo do systému DNS, takže by byly dostupné nástroji typu nslookup či dig, které slouží k vyhledávání a zobrazování informací z DNS serveru. Vložení by ale musel provést příslušný administrátor, což by představovalo jeho dodatečnou zátěž. Možná i proto se podpora a využívání této metody příliš nerozšířily a pokud ano, pak většinou pro popis poloh významných zařízení typu servery nebo směrovače [2]. Struktura sytému DNS je hierarchická. Díky tomu lze z DNS záznamu často určit například zemi ve které se hledaná stanice nachází [1].

Na Obr. 1.3 je zobrazena odpověď dotazu opět na doménu vutbr.cz. V jeho vrchní části je zobrazena část odpovědi nástroje z internetových stránek network-tools.com [6], spodní část Obr. 1.3 pak obsahuje odpověď na dotaz na stejnou doménu z internetových stránek dig-nslookup.nmonitoring.com [7].

name	class	type	data		time to live
vutbr.cz	IN	SOA	server:	rhino.cis.vutbr.cz	86400s (1d)
			email:	slama@vutbr.cz	
			serial:	2014050901	
			refresh:	28800	
			retry:	7200	
			expire:	604800	
			minimum ttl:	86400	

Name	Type	IP address	Class	TTL
vutbr.cz	NS	pipit.cis.vutbr.cz	IN	3805
vutbr.cz	NS	rhino.cis.vutbr.cz	IN	3805
vutbr.cz	MX	mail.vutbr.cz	IN	32108

Obr. 1.3 Prohledávání DNS záznamů

1.1.3 Použití dodatečných zdrojů informací

Tato myšlenka zajišťuje poměrně vysokou přesnost zjištění polohy. Zdroje dodatečných informací nemusí být zdaleka vždy dostupné a navíc jsou vhodné pouze pro určité situace, například pro zařízení využívající bezdrátové připojení, jak bylo zmíněno v úvodu této podkapitoly (Wifi, GSM). Je-li stanice v dosahu dostatečného počtu přístupových bodů Wifi nebo základnových stanic GSM, lze její polohu určit obdobou triangulace, což je způsob zjišťování souřadnic a vzdáleností trigonometrickým výpočtem. Místo měření úhlů se však použijí údaje o intenzitě elektromagnetického pole generovaného přístupovým bodem, o časovém zpoždění (době odezvy) apod. Zásadní problém této metody je ovšem potřeba přístupu k informacím získaných těmito přístupovými body nebo základnovými stanicemi [2].

Wifi sítě

Při použití Wifi sítě se k lokalizaci používá internetový prohlížeč. Ten při lokalizování stanice načte informace o všech přístupových bodech ve svém okolí jako SSID (Service Set Identifier), MAC (Media Access Control) adresu a sílu signálu a odešle je k porovnání s databází GoogleMaps. Tato databáze obsahuje přístupové body Wifi, kde je zaznamenána jejich MAC adresa, SSID a předpokládaná geografická poloha. Poloha přístupových bodů se počítá na základě polohy jiných přístupových bodů, které jsou předem známy a nachází se v okolí hledaného přístupového bodu. Pokud databáze GoogleMaps obdrží žádost na lokalizaci stanice spolu se seznamem přístupových bodů v okolí hledané stanice, vyhledá tyto přístupové body ve své databázi a na základě jejich vzájemné polohy a úrovní signálů změřených hledanou stanicí pak spočítá místo, kde se hledaná stanice nachází s největší pravděpodobností. Databáze přístupových bodů GoogleMaps je plně automatizovaná a dokáže si sama opravovat a aktualizovat záznamy [1], [3].

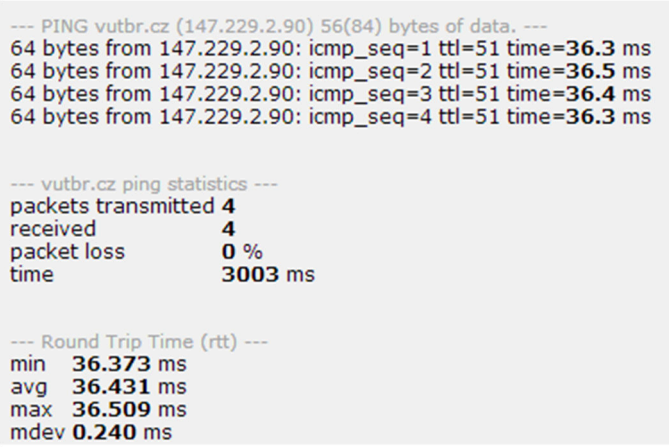
1.2 Aktivní geolokační metody

Princip většiny aktivních metod je založen na korelaci mezi zpožděním a geografickou vzdáleností. To je dáno vlivem zpoždění způsobeného rychlostí šíření signálu v médiu, které je hlavní složkou zpoždění na dlouhých vzdálenostech. Nejčastěji se měří zpoždění a zjišťuje se cesta dat přes síťové uzly od zdroje těchto dat k cílové stanici.

Zpoždění je doba potřebná na přenos jednoho datového segmentu od zdroje k příjemci. Obecně může být zpoždění ovlivněno mnoha faktory, jako jsou například přenosová rychlost vedení, momentální zatížení vedení, geografická vzdálenost zdrojové a cílové stanice, případně výkonností a aktuálním zatížením mezilehlých uzlů. Těmi jsou nejčastěji směrovače a přepínače připojené v síti na cestě od zdrojové stanice k cílové stanici. Podle místa, kde zpoždění v síti vzniká, jej lze dělit na:

- zpoždění na koncových zařízeních,
- zpoždění na mezilehlý uzlech,
- zpoždění na přenosových linkách.

Většinou se k lokalizaci jedné stanice používá větší množství referenčních bodů (řádově desítky). Existuje mnoho nástrojů na měření přenosového zpoždění a hledání cesty od zdroje k cíli. Mezi nejzákladnější a nejpoužívanější patří nástroj ping, který odešle dotaz na vzdálenou stanici a čeká na její odpověď - tím je změřena doba přenosu ke vzdálené stanici a zpět. Tato změřená hodnota se nazývá RTT (Round-Trip Time). Zpoždění se měří v obou směrech, protože cesta vysílaného signálu může být jiná od zdroje k cíli, než cesta signálu v opačném směru. Ukázka výstupu nástroje ping je uvedena na Obr. 1.4 pro doménu vutbr.cz z internetové adresy ping.eu [8].



Obr. 1.4 Výstup nástroje ping

Dalším nejzákladnějším nástrojem je nástroj traceroute, který zjistí IP adresy stanic mezi komunikujícími stranami. Oba výše zmíněné nástroje pracují na třetí vrstvě referenčního modelu OSI (Open Systems Interconnection) a využívají ke své činnosti protokol ICMP (Internet Control Message Protocol). Ukázka výstupu nástroje traceroute je uvedena na Obr. 1.5 pro doménu vutbr.cz z internetové adresy ping.eu [8].

traceroute to vutbr.cz (147.229.2.90), 30 hops max, 60 byte packets

1	static.121.168.4.46.clients.your-server.de	46.4.168.121	de	**2.503** ms	**2.581** ms	**2.774** ms
2	hos-tr3.juniper2.rz13.hetzner.de	213.239.224.65	de	**0.149** ms		
	hos-tr1.juniper1.rz13.hetzner.de	213.239.224.1	de	**0.340** ms		
	hos-tr3.juniper2.rz13.hetzner.de	213.239.224.65	de	**0.149** ms		
3	core22.hetzner.de	213.239.245.121	de	**0.261** ms	**0.259** ms	**0.257** ms
4	core11.hetzner.de	213.239.245.225	de	**2.808** ms		
	core11.hetzner.de	213.239.245.221	de	**2.807** ms	**2.804** ms	
5	juniper4.rz2.hetzner.de	213.239.203.138	de	**2.821** ms	**2.795** ms	**2.813** ms
6	ams-ix-1.cesnet.cz	195.69.145.106	nl	**32.848** ms	**32.846** ms	**32.877** ms
7	gw-ant.net.vutbr.cz	147.229.252.18	cz	**37.667** ms	**37.871** ms	**38.066** ms
8	pe-ant.net.vutbr.cz	147.229.253.236	cz	**37.009** ms	**36.785** ms	**37.242** ms
9	irf-ant.net.vutbr.cz	147.229.252.206	cz	**37.979** ms	**37.549** ms	**37.759** ms
10	piranha.ro.vutbr.cz	147.229.2.90	cz	**36.526** ms		

Obr. 1.5 Výstup nástroje traceroute

1.2.1 GeoPing

Tato metoda je založená na měření zpoždění a z metod založených na tomto principu je nejstarší. Pro co nejpřesnější fungování metody je potřeba velké množství pasivních referenčních bodů. Těmto bodům (uzlům) se také říká landmarky a jejich geografická poloha je známa. Dále je potřeba několika aktivních sond (uzlů provádějících měření). Nevýhoda této metody spočívá v určení výsledné polohy jako místa, kde leží jeden z referenčních bodů. Tímto je omezena přesnost metody a proto je důležité disponovat co největším počtem referenčních bodů, jak bylo zmíněno výše.

Referenční body by měly být geograficky rovnoměrně rozloženy a připojeny spolehlivým vysokorychlostním spojem. Aktivních sond je doporučeno 7-9. Ty dokáží změřit dobu zpoždění k jednotlivým referenčním bodům a cílové stanici. I tyto sondy by měly být geograficky rovnoměrně rozmístěny. Princip metody je založen na porovnávání vektorů zpoždění příslušejících referenčním bodům a lokalizované stanici. Vektor zpoždění obsahuje změřenou dobou přenosu informace mezi referenčním bodem a všemi sondami. Stejný vektor je změřen také pro lokalizovanou stanici. Ten je pak porovnán s vektory referenčních bodů. Pro nalezení nejpodobnějšího vektoru je pak vytvořen N-rozměrný prostor. V tomto prostoru je nalezen vektor s nejmenší euklidovskou vzdáleností k hledanému vektoru podle

vztahu (1.1), kde symboly d_1 až d_N představují minimální zpoždění mezi referenčním bodem a sondami a d'_1 až d'_N jsou minimální zpoždění mezi lokalizovanou stanicí a sondami [1], [3].

$$DV' = \sqrt{(d_1 - d'_1)^2 + \cdots + (d_N - d'_N)^2} \qquad (1.1)$$

1.2.2 ShortestPing

Tato metoda je principiálně nejjednodušší. Stejně jako u metody GeoPing i zde je požadavek pro co největší počet rovnoměrně rozprostřených referenčních bodů, u kterých je známá geografická poloha. Opět se zde zjišťuje zpoždění mezi lokalizovanou stanicí a všemi referenčními body a jako výsledná poloha je pak určena poloha referenčního bodu, ke kterému bylo změřeno nejmenší zpoždění. I přes svoji jednoduchost může tato metoda dosáhnout lepších výsledků, než například metoda GeoPing. V některých případech však může být skutečná pozice mnohem vzdálenější, než pozice vybraného referenčního bodu [1], [3].

1.2.3 Constraint Based Geolocation (CBG)

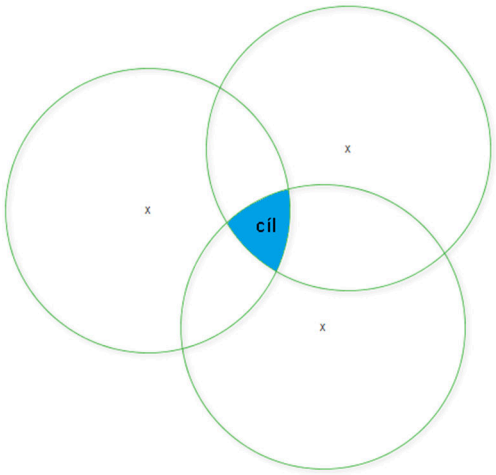
Metoda CBG ke své činnosti využívá trilaterace známé z rádiového určování polohy a pro svou činnost požaduje množinu aktivních referenčních bodů se známou polohou. Její princip spočívá ve využití vztahu mezi geografickou vzdáleností a zpožděním, čímž vznikne tzv. hranice nejvzdálenějšího možného umístění stanice. Tato hranice je určena přepočtem zpoždění na základě tzv. bestline. To je přímka vytvořená při kalibraci a udává vztah mezi zpožděním a vzdáleností pro příslušný referenční bod. Prvním krokem je tedy kalibrace.

Dalším krokem je samotná lokalizace. Zde změří referenční bod zpoždění k cílové stanici a toto zpoždění se pak přepočítá na vzdálenost podle obecného vztahu (1.2), kde k je směrnice přímky a b_i je posunutí na ose y.

$$vzdálenost = k \cdot zpoždění + b_i \qquad (1.2)$$

Tato vzdálenost udává zmíněnou hranici (poloměr kružnice), za kterou se cíl už pravděpodobně nenachází. Cílová pozice stanice je pak určena průnikem kružnic jednotlivých referenčních bodů a nalezením těžiště této oblasti průniku. Jednoduchá ukázka pro 3

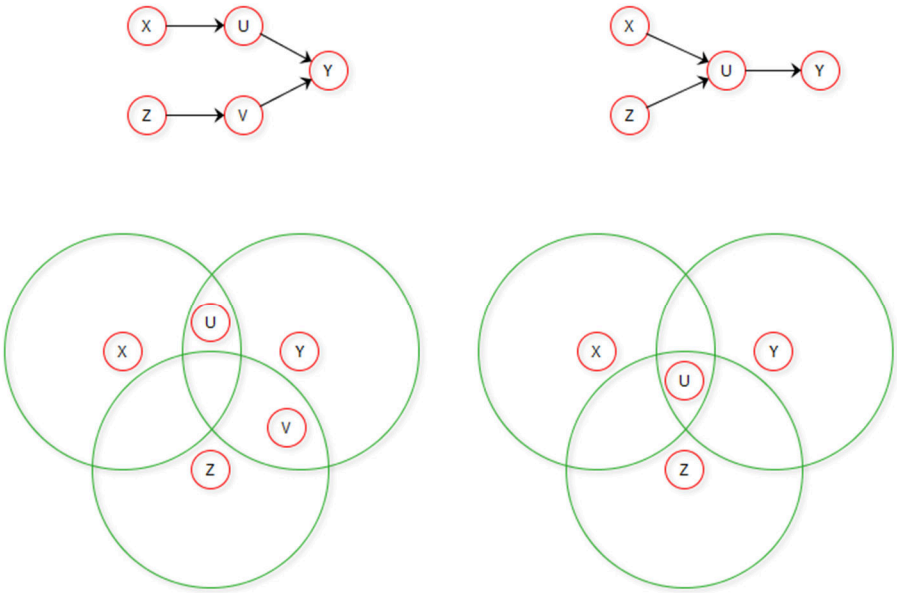
referenční body je uvedena na Obr. 1.6. Velikost průniku určuje také chybovou oblast, ve které se cílová stanice může nacházet [1], [3].



Obr. 1.6 Metoda CBG

1.2.4 Topology Based Geolocation (TBG)

Metoda TBG je rozšířená metoda CBG. Zde se pří měření zpoždění využívá informace o topologii a směrování v síti, přes kterou data prochází. Metoda se snaží odhadnout pozici cílové stanice na základě měření zpoždění od jednotlivých referenčních bodů a také odhadované pozice mezilehlých uzlů [1], [9]. Princip metody je zobrazen na Obr. 1.7, kde v závislosti na topologii je hledán cílový uzel.



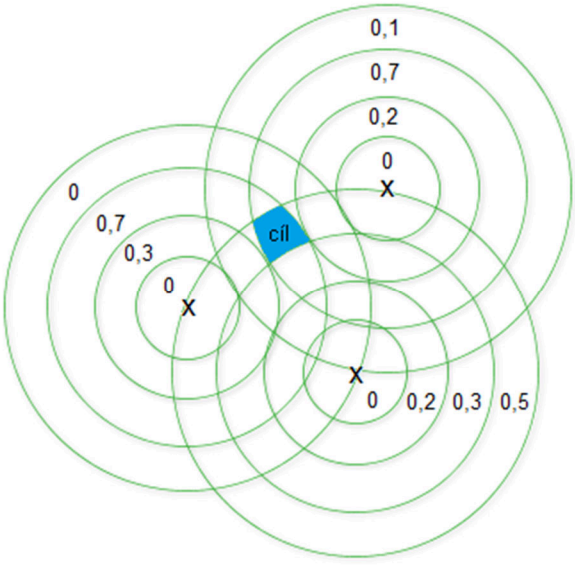
Obr. 1.7 Metoda TBG

1.2.5 Speed of Internet (SOI)

Metoda SOI vychází z metody CBG. Opět zde dochází k vytváření hranice, za kterou se cíl už nemůže nacházet. Je zde tedy také potřeba aktivních referenčních bodů se známou polohou. Rozdíl oproti CBG je v počítání vzdálenosti ze zpoždění, kdy je použita konstanta $^{4}/_{9} \cdot c$, kde c je rychlost světla, namísto přímky vypočítané z dat získaných kalibrací [9]. SOI tedy nepotřebuje kalibrační měření. Tím je zmenšena zátěž sítě. Nevýhodou je však menší přesnost, protože oblast průniku kružnic je větší. Existuje ale také možnost, že se kruhy neprotnou vůbec [3].

1.2.6 Geoweight

GeoWeight je opět metoda založená na principech metody CBG. Stejně jako u CBG i zde je nutné provést kalibraci. Kalibrací jsou získány zpoždění mezi referenčními body a poté jsou vytvořena rovnoměrná pásma vzdáleností, kterým jsou přiřazena odpovídající naměřená zpoždění. Podle počtu přiřazených zpoždění je každé vzdálenosti přidělena odpovídající pravděpodobnost, tedy váha. Při lokalizaci cíle jsou pak změřenému zpoždění ke stanici přiřazena pásma vzdáleností a pravděpodobnost, kde se cíl může vyskytovat. Výsledná pozice je opět určena průnikem oblastí [3], [10]. Tentokrát se však berou v potaz protnutá mezikruží s nejvyšším součtem pravděpodobností, jak je pro ukázku uvedeno na Obr. 1.8.



Obr. 1.8 Metoda GeoWeight

1.2.7 Spotter

Stejně jako předchozí tři popsané metody i tato metoda vychází z metody CBG. Data získaná kalibrací jsou zde podrobena statistické analýze a na tomto základě je vytvořeno normální (Gaussovské) rozdělení pravděpodobnosti vzdáleností. Následně je změřeno zpoždění d_i od referenčního bodu k cílové stanici a podle něj je vytvořena kružnice se středem v daném referenčním bodu. Na blízkém okolí této kružnice je pak definována hustota pravděpodobnosti dle zkalibrovaného Gaussova rozdělení. Oblast vysoké pravděpodobnosti výskytu cílové stanice je zjištěna průnikem těchto hustot pravděpodobností od všech referenčních bodů [3], [11].

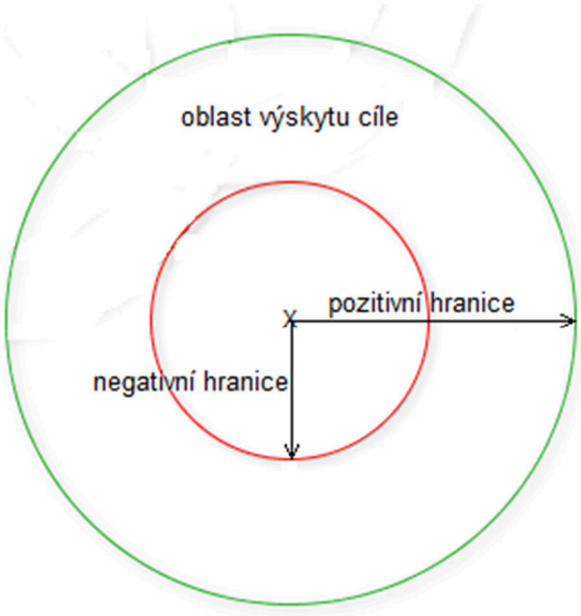
1.2.8 Octant

Poslední popisovaná metoda bude metoda Octant, která je předmětem této práce a vysvětlení jejího principu bude věnována celá následující kapitola. Nejdříve bude princip této metody popsán z širšího pohledu a poté už bude následovat podrobnější popis v podkapitolách krok po kroku.

1.3 Geolokační metoda Octant

1.3.1 Obecný popis metody

Většina výše popsaných metod měla jednu věc společnou. Jejich princip vycházel z metody CBG, jejíž hlavní myšlenkou je vytvoření hranice omezení, za kterou už s největší pravděpodobností hledaný cíl neleží. Ta je označována jako pozitivní hranice. Funkčnost metody Octant je také založená na těchto hranicích, jen je zde přidána navíc i negativní hranice, která udává vzdálenost, před kterou hledaný cíl neleží. Jak již bylo popsáno dříve, tyto hranice jsou vlastně poloměry kružnic se středem v referenčním bodu, kterému tyto hranice patří, viz Obr. 1.9. Tyto poloměry jsou získány přepočtem změřeného zpoždění od referenčního bodu k cílovému bodu na geografickou vzdálenost. Pro mapování zpoždění na vzdálenost se obdobně jako u CBG používá kalibračních dat mezi referenčními body. K přepočtu je zde však využita konvexní obálka všech změřených dat. Spodní strana této obálky je pro určení pozitivní hranice a horní strana pro určení negativní hranice [3]. Postup při vytváření obálky je popsán v další podkapitole.



Obr. 1.9 Hranice omezení u metody Octant

1.3.2 Zjištění hranic omezení (kalibrace)

Naplnění korelačních tabulek

Jak již bylo zmíněno v úvodním popisu metody, před lokalizací je nutné, aby si každý referenční bod vytvořil své hranice omezení. První krok, který musí referenční bod provést, je odeslání ICMP dotazu ke všem ostatním referenčním bodům pro získání zpoždění v síti. Po provedení tohoto kroku mají tedy všechny referenční body informace o zpoždění k těm ostatním. Tyto hodnoty si uloží do svých tabulek, kterým se říká korelační.

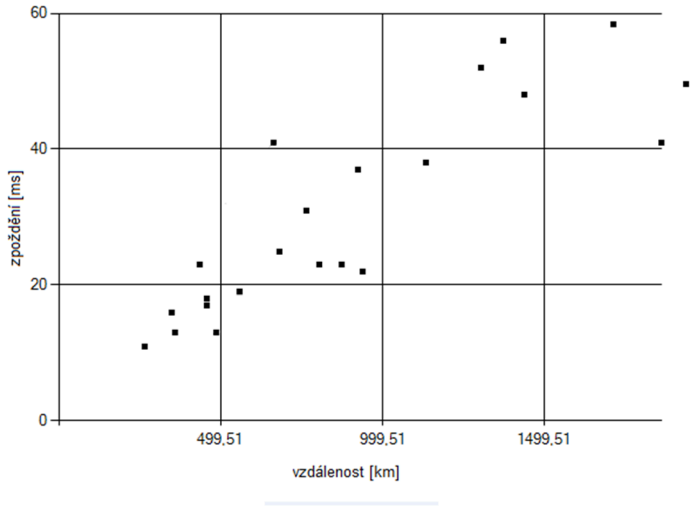
Hodnoty v korelačních tabulkách jsou v pravidelných intervalech aktualizovány a tím jsou ošetřeny změny v síti. Podle vzorce (1.3) je z hodnot poloh referenčních bodů vypočítána vzdálenost mezi sebou navzájem [13]. Písmeno *d* představuje vzdálenost, φ_1 a φ_2 jsou zeměpisné šířky poloh obou referenčních bodů, $\Delta\lambda$ pak označuje rozdíl zeměpisných délek poloh obou referenčních bodů. Poslední neznámé písmeno *R* značí poloměr zeměkoule, jehož velikost je 6372,8.

$$d = Acos(sin(\varphi_1) \cdot sin(\varphi_2) + cos(\varphi_1) \cdot cos(\varphi_2) \cdot cos\Delta\lambda)) \cdot R \quad (1.3)$$

Vzdálenost si spočítá každý referenční bod a uloží ke stejnému uzlu do své korelační tabulky. Nyní tedy každá korelační tabulka každého referenčního bodu obsahuje hodnoty zpoždění i vzdálenosti pro ostatní uzly.

Vytvoření horní a dolní obálky

V tomto okamžiku si lze hodnoty korelační tabulky představit jako graf bodů zpoždění/vzdálenost (Obr. 1.10), ze kterých je potřeba získat horní a spodní obálku.

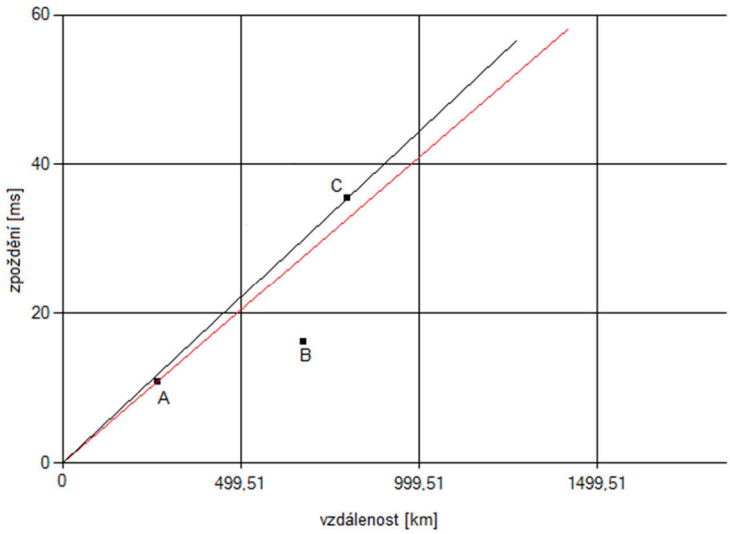


Obr. 1.10 Graf bodů zpoždění/vzdálenost

To se provede tak, že se všechny body v grafu seřadí vzestupně podle zpoždění. Začíná se od nulové hodnoty a od první seřazené hodnoty. Pro ně je spočtena směrnicová rovnice přímky ze vztahu (1.4), kde *k* označuje směrnici přímky, *x* a *y* jsou souřadnice bodů a *q* udává úsek, který přímka vytíná na ose *y* [14].

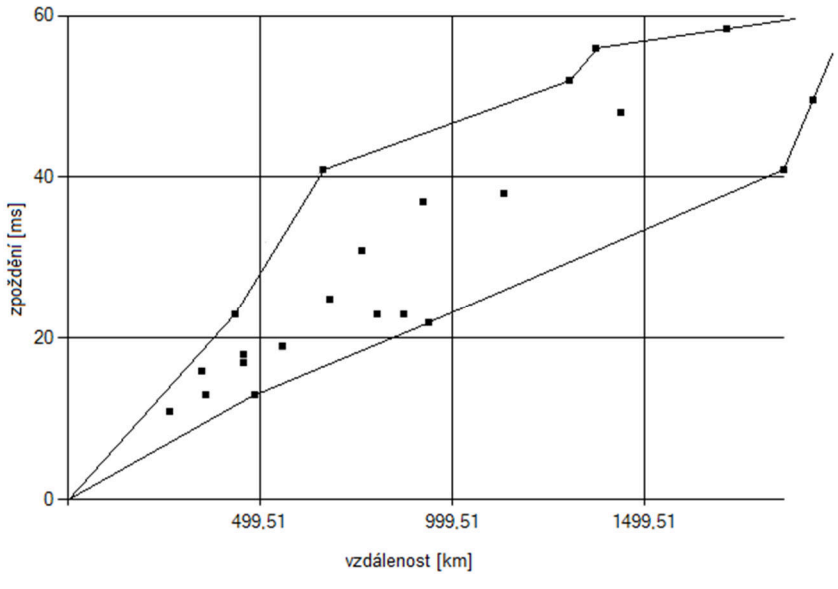
$$y = k \cdot x + q \quad k, q \in R \qquad (1.4)$$

Vytváření horní obálky je zobrazeno na Obr. 1.11. Zde dočasná přímka (zobrazena červeně) tedy vede z bodu 0 do prvního bodu (bod A) a zjišťuje se, jestli všechny ostatní body leží pod touto přímkou. To se zjistí tak, že jsou všechny body postupně dosazovány do spočtené směrnicové rovnice přímky. Pokud tato rovnice vyjde větší než nula pro daný bod, vše je v pořádku (v případě bodu B) a může být dosazován další bod. Pokud rovnice vyjde menší než nula, bod leží nad přímkou a musí nahradit aktuální bod (bod C nahradí bod A). Ve skutečnosti je v grafu bodů mnohem více a a proto se stejné operace provádějí pro tuto novou kombinaci bodů, dokud není podmínka všech bodů pod přímkou splněna.



Obr. 1.11 Vytváření obálky

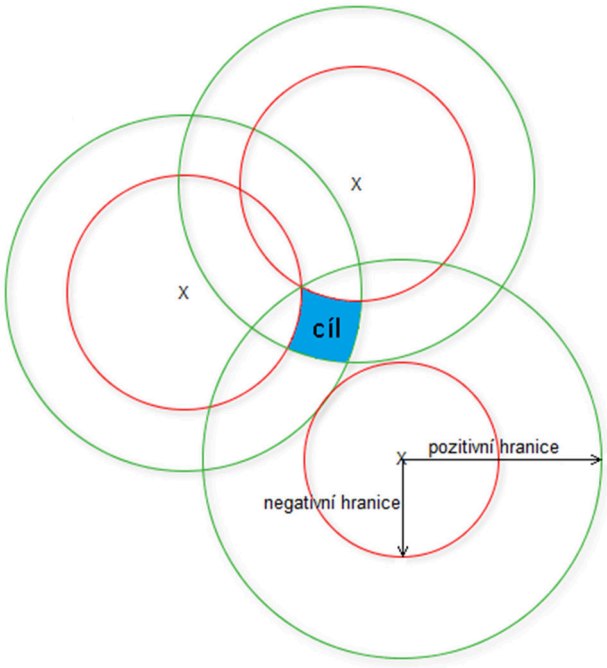
Po splnění podmínky všech bodů pod přímkou dostaneme první přímku obálky. Pro získání další přímky obálky už jsou další kroky analogické, jen se jako počáteční bod přímky vezme koncový bod získané první přímky. Vše se opakuje, dokud nezískáme celou horní obálku, tvořenou z vypočtených přímek. Pro získání dolní obálky se provádí totožné operace jako pro získání té horní, jen se změní porovnávání, kdy body musí ležet nad přímkou. Výsledné obálky pak budou pro Obr. 1.10 vypadat jako na Obr. 1.12.



Obr. 1.12 Horní a spodní obálka

1.3.3 Hledání cílové stanice (lokalizace)

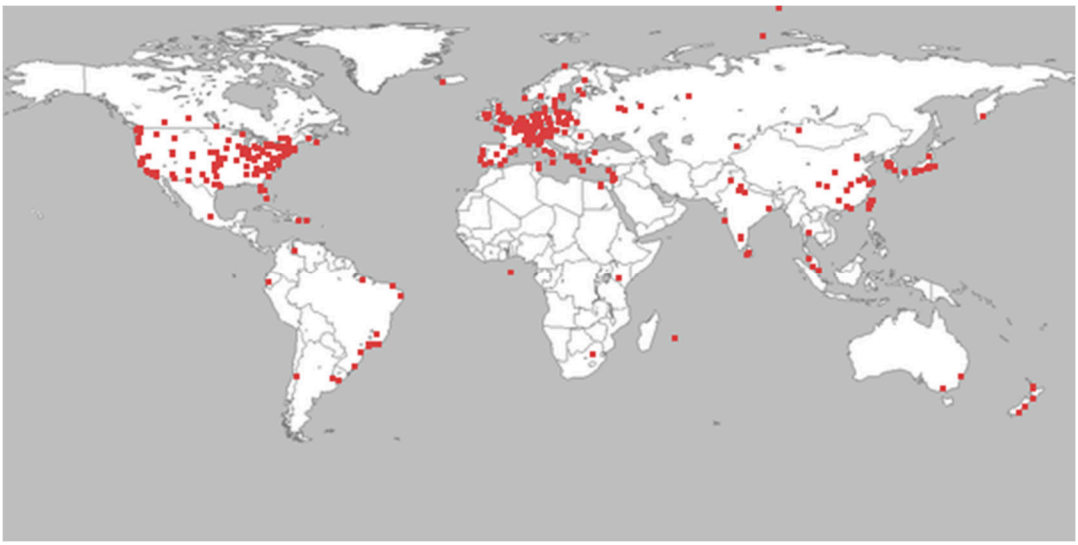
Nyní jsou vytvořeny obálky pro určení velikosti hranic omezení, takže může začít lokalizování hledané stanice. Každý referenční uzel musí zjistit zpoždění od hledané stanice, což provede opět ICMP dotazem. Získané zpoždění si uloží a najde pro něj odpovídající přímku v seznamu přímek pro horní obálku. To se provede tak, že se jen porovnává počáteční a koncový bod, lépe řečeno hodnota zpoždění z grafu pro daný bod s hledaným zpožděním. Pokud hledané zpoždění spadá do intervalu těchto dvou zpoždění, pak je přímka nalezena a do její rovnice se pak jen dosadí získané zpoždění a vypočte se z ní vzdálenost. Tato vzdálenost udává poloměr kružnice negativní hranice. Vše se pak opakuje pro seznam přímek pro dolní obálku, takže je po nalezení odpovídající přímky spočten poloměr kružnice pozitivní hranice. Tímto máme vypočteny potřebné hodnoty pro vynesení mezikruží. Cílová stanice, neboli její nejpravděpodobnější umístění, leží v mezikruží pozitivní a negativní hranice, ne dále a ne blíže, než jsou hodnoty těchto hranic. Největší hranice omezení mají ty uzly, které naměřily největší zpoždění k cíli. To znamená, že jsou od cílového uzlu vzdáleny nejdále. Protnutím mezikruží od nejvzdálenějších po nebližší se hledaná oblast začne omezovat a její velikost zmenšovat. Poloha cílové stanice je pak dána těžištěm plochy průniku těchto mezikruží od všech referenčních bodů, jak je zobrazeno na Obr. 1.13.



Obr. 1.13 Poloha cílové stanice

1.4 Experimentální síť PlanetLab

Experimentální síť PlanetLab je celosvětová počítačová síť, jejíž hlavní úlohou je umožnit uživatelům zkoumat a testovat síťové aplikace, protokoly a podobně. Vznikla v roce 2002 jako konsorcium několika amerických univerzit a časem se do ní přidaly i další. Jejími členy však nejsou jen univerzity, ale i řada výzkumných pracovišť významných firem v oblasti teleinformatiky a výpočetní techniky [15]. Do každého projektu je zařazeno několik uzlů. Tato množina uzlů se nazývá slice. Slice uživatelům přiřazuje tzv. PI (Principal Investigator), který je za každý vytvořený slice zodpovědný. Po přiřazení slice k uživateli může pak uživatel do nich tyto uzly přidávat. Struktura PlanetLabu je rozdělena na vrstvy, přičemž na nejnižší vrstvě jsou tyto jednotlivé stanice, na kterých běží linuxová distribuce, poskytující nezávislý virtuální stroj pro každého uživatele. Toto řešení umožňuje využívat prostředky jednoho uzlu více uživatelům současně. Uživatelé mohou být ke svým účtům připojeni buď přes příkazovou řádku v linuxovém systému, nebo využít programu PuTTY, pokud se připojují z operačního systému Windows. Připojení je řešeno pomocí protokolu SSH (Secure Shell). Ten zajišťuje šifrovanou komunikaci se vzdáleným uzlem a také autentizaci pomocí veřejného klíče. Uzly se společnými vlastnostmi (např. patřící jedné organizaci) se řadí do tzv. sites [16], [17]. Rozložení PlanetLab uzlů po světě je zobrazeno na Obr. 1.14 (převzat z [18]).



Obr. 1.14 Rozložení uzlů PlanetLab

1.5 Programovací jazyk C#

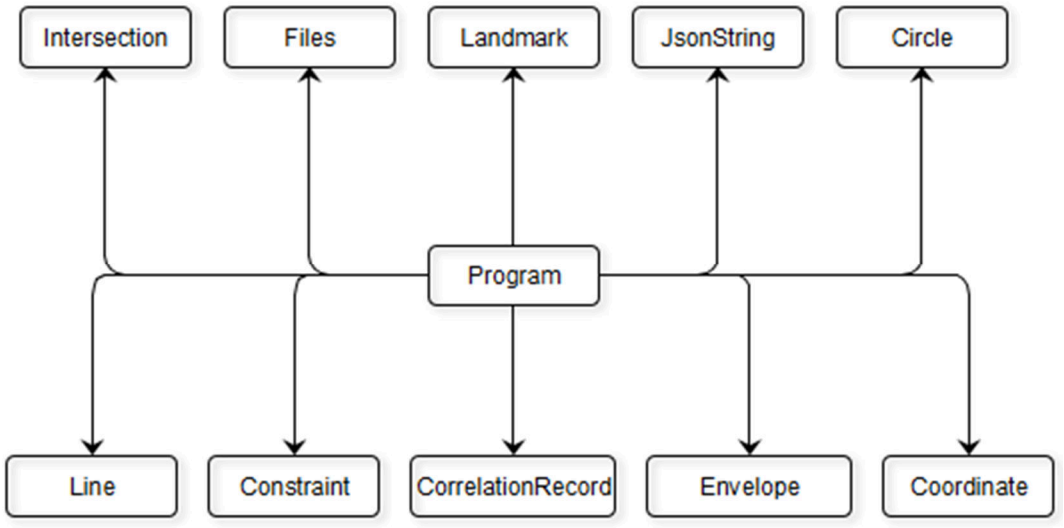
Protože bude celý projekt realizován v programovacím jazyce C#, je vhodné jej alespoň představit a základně charakterizovat. Jazyk C# byl vyvinut firmou Microsoft a uveden spolu s vývojovým prostředím .NET. Vychází z programovacího jazyka C/C++, ale je daleko bližší jazyku Java. Jeho základní charakteristiky jsou [19]:

- Jazyk C# je čistě objektově orientovaný.
- Obsahuje nativní podporu komponentového programování.
- Podobně jako Java obsahuje pouze jednoduchou dědičnost s možností násobné implementace rozhraní.
- Vedle členských dat a metod přidává vlastnosti a události.
- Správa paměti je automatická. O korektní uvolňování zdrojů se stará tzv. garbage collector.
- Podporuje zpracování chyb pomocí vyjímek.
- Zajišťuje typovou bezpečnost a podporuje řízení verzí.
- Podporuje atributové programování.
- Zajišťuje zpětnou kompatibilitu se stávajícím kódem jak na binární, tak na zdrojové úrovni.

2 ZKOUMÁNÍ PŘESNOSTI IP GEOLOKACE POMOCÍ GEOLOKAČNÍ METODY OCTANT

V průběhu realizace metody Octant bylo potřeba řešit mnoho dílčích podproblémů. Metoda byla naprogramována v jazyce C# a bylo tedy potřeba navrhnout a vytvořit třídy, ze kterých se program skládá. Použité třídy při realizaci metody Octant jsou zobrazeny na Obr. 2.1. V dalších podkapitolách pak budou funkce těchto tříd popsány podrobněji v pořadí, ve kterém byly vytvářeny a podle jejich složitosti. Každá třída mimo metod a proměnných samozřejmě obsahuje i konstruktory pro danou třídu, ty ale není potřebné popisovat.

Projekt byl naprogramován jako konzolová aplikace, protože v konečné fázi program poběží v linuxovém příkazovém řádku. Dále byl vytvořen jednoduchý program opět v jazyce C#, tentokrát jako formulářová aplikace, aby bylo možné ilustrovat výsledky a mezivýsledky programu (kružnice, průsečíky) graficky. Popis jeho funkce však není důležitý, slouží jen pro zpracování diplomové práce.



Obr. 2.1 Použité třídy při realizaci metody Octant

2.1 Struktura programu geolokační metody Octant

2.1.1 Třída Coordinate

Tato třída byla vytvořena jako první a tvoří základ většině tříd, které z této třídy vychází. Obsahuje pouze dvě hlavní proměnné, ale o to víc jsou důležité. Jsou jimi proměnné AxisX a AxisY pro uložení souřadnic objektu. Dále třída už obsahuje metody, které se starají o různé operace. Jsou to:

- **countPointsDistance** pro spočítání vzdálenosti mezi dvěma body,
- **pointsOverLine** a **pointsUnderLine** pro použití při výpočtu obálek a rozhodování, zda bod leží nad přímkou či pod ní,
- **sortByAxisX** pro seřazení objektů podle souřadnice x, dále
- **countDownEnvelope** a **countUpEnvelope** pro výpočet obálek, obě tyto metody tedy vrací seznam přímek, ze kterých se obálka skládá a dvě poslední metody
- **convertDegreeToDistance** a **convertDistanceToDegree** pro převod stupňů zeměpisných šířek a délek na vzdálenosti a zpět.

Poslední dvě jmenované funkce jsou důležité z toho hlediska, že na počátku jsou zadány všechny souřadnice ve formátu zeměpisné šířky a délky a aby se s nimi dalo dále pracovat, je potřeba je převést na číslo odpovídající kilometrové vzdálenosti. Při převodu stupňů na vzdálenost se vynásobí zeměpisná šířka konstantou 111,330 a zeměpisná délka konstantou 64,152. Tento převodní poměr platí, pokud je střed Evropy umístěn v Litvě na souřadnicích 54,9° a 23,32° [20]. Při převodu zpět na stupně se samozřejmě obě hodnoty odpovídajícími konstantami zase vydělí.

2.1.2 Třída Line

Třída Line je o poznání méně obsáhlá než předchozí třída, ale obsahuje vše potřebné pro reprezentaci přímky. Jsou to tedy proměnné k a q pro směrnici přímky a velikost jejího posunutí na ose y a dvě proměnné pointFirst a pointEnd typu Coordinate pro reprezentaci počátečního a koncového bodu přímky. Dále třída obsahuje dvě metody. Jsou jimi:

- **countLine** pro spočtení přímky pro dva body,
- **findLine** pro nalezení odpovídající přímky při výpočtu hranic omezení pro daný cíl, kdy se hledá interval, do kterého spadá hodnota zpoždění od cíle, neboli přímka pro toto zpoždění.

2.1.3 Třída CorrelationRecord

Tato třída reprezentuje záznam v korelační tabulce každého referenčního uzlu. Hlavní proměnné jsou tři. IpAddress pro uložení IP adresy, Delay pro zpoždění a Distance pro vypočtenou vzdálenost pro uložený vzdálený referenční bod. Používané (můžou se zde vyskytovat i metody, které se využívaly při vývoji, ale v konečném projektu využity nebyly) metody v této třídě jsou dvě a jsou to:

- **deleteCorrelationRecord** pro smazání záznamu z tabulky aktuálního uzlu (například při nesprávné velikosti hodnoty zpoždění) a
- **findIndexOfRecordInCorrelationTable**, která souvisí s předchozí metodou a je potřebná pro nalezení správného záznamu, protože deleteCorrelationRecord pracuje pouze s IP adresou uzlu určeného k vymazání z korelační tabulky.

Smazání záznamů z korelačních tabulek je provedeno vždy na obou stranách. Nejdříve je tedy smazán záznam z korelační tabulky uzlu, který špatné zpoždění detekoval pro danou IP adresu a poté je smazán záznam i u vzdáleného uzlu pro aktuální IP adresu uzlu, který smazání inicioval.

2.1.4 Třída Constraint

Tato třída je obsahově velmi málo rozsáhlá. Používá se jen pro uložení hodnot negativního a pozitivního poloměru u hranic omezení. Obsahuje celkem jen tři proměnné. Jsou to RadiusNegative a RadiusPositive pro dané hranice a proměnná succeess pro uložení úspěšnosti výpočtu těchto hranic.

2.1.5 Třída Envelope

Využití této třídy je jen pro uložení dvou bodů a přímky. Třída tedy nemá žádné metody, má jen tři objekty a jsou jimi dva objekty Coordinate s názvy PointFirst a PointEnd a jeden objekt Line s názvem Line.

2.1.6 Třída Files

Třída se stará o správu souborů. Obsahuje tedy jen metody pro zápis/čtení hodnot do/ze souboru a jsou jimi

- **writeTextToFile** pro zápis právě prováděné činnosti do souboru. Za zmínku zde stojí jen ověřování, zda již byl soubor vytvořen, aby nedocházelo k neustálému zakládání nových souborů, ale informace byly zapisovány do aktuálního souboru. Další metody jsou
- **saveEnvelopeToFile** a **readEnvelopeFromFile** pro ukládání/načtení obálek do/ze souboru a jako poslední metody
- **saveConstraintsToFile** a **readConstraintsFromFile** pro uložení a načtení obálek do/ze souboru.

2.1.7 Třída Landmark

Třída Landmark je jedna ze dvou nejobsáhlejších tříd z hlediska proměnných, objektů a metod, protože tvoří jádro celého projektu. Ke své funkčnosti využívá všech předchozích tříd. Třída Landmark je zděděna ze třídy Coordinate. Má tedy proměnné pro uložení svých souřadnic a dále přidává proměnné IpAddress pro uložení IP adresy, seznam objektů CorrelationRecord zvaný correlationTable, dále seznamy přímek pro dolní a horní obálku nazvaných downEnvelope a upEnvelope, objekt Constraint pro hranice omezení, proměnnou DelayFromTarget pro uložení zpoždění od hledaného cíle a dvě pravdivostní proměnné. Proměnnou sshFailed pro zamezení opětovného připojování k uzlu, ke kterému připojení dříve selhalo a logCreated pro indikaci úspěšného vytvoření souboru u daného referenčního bodu, takže při opětovném SSH připojení k tomuto referenčnímu bodu kvůli čtení dříve vytvořeného souboru se jde „na jistotu“.

Jak již bylo řečeno, třída obsahuje nejvíce metod ze všech tříd. Nutno dodat, že metody, které se připojují přes SSH na referenční uzel pro provedení nějaké operace, jsou spouštěny paralelně pomocí vláken, takže jsou spuštěny všechny najednou, což zkrátí časovou náročnost provedení metody. Některé metody byly implementovány při vývoji a v konečném projektu nejsou využity. Ty použité jsou:

- **connectToLandmarkPingOthersSaveToLog**, která se přes SSH připojí na každý referenční bod (pokud je to možné), u něj smaže soubor vytvořený při předchozím připojení (pokud existuje), poté odešle ICMP dotaz na všechny ostatní referenční body, získané odpovědi odfiltruje podle potřeby a hodnoty zpoždění uloží u daného referenčního bodu do souboru. Je zde taky nastavena proměnná sshFailed a logCreated podle výsledku dané operace. Další metoda třídy Landmark je
- **connectToLandmarkReadLog**, která se opět připojí přes SSH k referenčním bodům a na základě stavu proměnné logCreated přečte obsah souboru vytvořeného předchozí metodou, tedy získané hodnoty zpoždění od ostatních referenčních bodů. Tento obsah rozčlení do potřebných proměnných a ty pak mohou být uloženy do korelačních tabulek. Další metodou je metoda
- **countDistance.** Ta má za úkol spočítat vzájemnou vzdálenost dvou referenčních bodů podle vzorce 1.4.2. Následující metoda
- **checkValidityOfDelayFromLandmarksCountDistance**, jak sám název napovídá, má za úkol zkontrolovat, zda uložená zpoždění v korelačních tabulkách všech referenčních bodů mají správnou velikost. Zadané kritérium je, že zpoždění musí být menší než 200ms a zároveň větší než 1ms. Pokud tato podmínka neplatí, je daný záznam vymazán z korelační tabulky metodou deleteCorrelationRecord ze třídy CorrelationRecord. Pokud je podmínka splněna, je k dané hodnotě zpoždění uložena hodnota vzdálenosti spočtené předchozí metodou countDistance pro odpovídající referenční bod, kterému zpoždění patří. Pátá metoda třídy Landmark je
- **connectToLandmarkPingTarget**, kde opět probíhá SSH připojení na referenční bod, proběhne ICMP dotaz pro zpoždění, hodnota odpovědi je odfiltrována jen na hodnotu zpoždění, dále je zkontrolována jeho velikost a v případě úspěšné kontroly si aktuální referenční bod tuto hodnotu zpoždění od cíle uloží. Předposlední metoda této třídy se nazývá

- **countConstraints.** Ta už provádí výpočet negativní a pozitivní hranice omezení. Nejdříve jsou ze souboru načteny přímky tvořící horní obálku do seznamu. Poté je použita funkce findLine třídy Line pro nalezení přímky pro dané zpoždění. V případě úspěchu je vypočtena negativní hranice a stejný postup je zopakován pro nalezení přímky v dolní obálce a spočtení pozitivní hranice. Poslední metoda této třídy je metoda
- **findIndexOfLandmark** pro nalezení indexu referenčního bodu podle zadané IP adresy v seznamu referenčních bodů. Této metody je využito při mazání korelačních záznamů a také ji využívá metoda pro validaci hodnot zpoždění od ostatních referenčních bodů checkValidityOfDelayFromLandmarksCountDistance.

Nutné je zmínit důležitý příkaz metody `connectToLandmarkPingOthersSaveToLog`, pomocí kterého aktuální referenční bod vyšle ICMP dotaz k dalšímu referenčnímu bodu pro zjištění zpoždění a odfiltrovanou odpověď uloží do souboru. Příkaz vypadá takto:

```
"IP=" + landmarks[i].IpAddress + "; DLY=`ping -q -i 0.5 -c 4 " + landmarks[i].IpAddress + " | grep rtt | sed -e 's/.*mdev = \\([^\\/]*\\).*/\\1/'`; echo $IP $DLY >> " + localIP + ".log;";
```

Hodnota proměnné IP označuje IP adresu právě pingovaného uzlu a do hodnoty DLY je uloženo odfiltrované získané zpoždění. Obě tyto proměnné jsou pak uloženy do souboru s názvem IP adresy lokálního počítače. Metody `connectToLandmarkReadLog` a `connectToLandmarkPingTarget` používají stejné odfiltrování odpovědi ICMP.

2.1.8 Třída Circle

Tato třída slouží pro vytváření objektů kružnic. Obsahuje tři proměnné, jsou to objekt Coordinate center, objekt Constraint s názvem constraint a proměnnou Radius pro reprezentaci poloměru kružnice. Třída obsahuje pouze jednu metodu a je jí

- **countMiddleRadiusesOfCircles**, která spočte střední poloměry kružnic z negativních a pozitivních poloměrů tak, že přičte negativní poloměr k rozdílu pozitivního a negativního poloměru děleném hodnotou 1,5 [20].

2.1.9 Třída Intersection

Předposlední popisovaná třída je třída Intersection, pomocí které jsou vytvářeny objekty průsečíků. Obsahuje nejvíce metod, ale není potřeba je všechny podrobně vysvětlovat. Tato třída je děděna ze třídy Coordinate a žádné další proměnné zde nejsou třeba. Metody v této třídě jsou

- **findIntersectionsOfMiddleCircles** pro nalezení průsečíků středních kružnic,
- **findIntersectionsInEveryCircle** pro nalezení průsečíku, nacházejícího se ve všech kružnicích, lépe řečeno musí být jimi obklopen,
- **countTargetIntersectionFromTwoIntersections** pro výpočet polohy cílového průsečíku ze dvou průsečíků kružnic. Cílový průsečík, lépe řečeno jeho souřadnice x je vypočítán jen jako sečtení souřadnic osy x dvou průsečíků kružnic a podělení tohoto součtu dvěma. Stejně je spočtena souřadnice y pro cílový průsečík. Další metoda je
- **countTargetIntersection** sloužící pro výpočet cílového průsečíku, k tomu používá tři následující funkce. První je
- **countArea** pro výpočet oblasti cílového průsečíku podle vzorce (2.1) [21], kde A je počítaná oblast, x_n a y_n jsou souřadnice průsečíků a N je jejich celkový počet. Další metoda je
- **countAxisxOfTargetIntersection** pro výpočet souřadnice x cílového průsečíku a
- **countAxisyOfTargetIntersection** pro výpočet souřadnice y cílového průsečíku.

$$A = \frac{1}{2}\sum_{n=0}^{N-1}\begin{vmatrix} x_n & x_{n+1} \\ y_n & y_{n+1} \end{vmatrix} \qquad (2.1)$$

Další implementované metody slouží jen pro mezivýpočty při výpočtu cílového průsečíku, za zmínku stojí výpočet vzorců (2.2) a (2.3), kde c_x a c_y jsou souřadnice cíle [21].

$$c_x = \frac{1}{6A}\sum_{n=0}^{N-1}\left(x_n + x_{n+1}\begin{vmatrix} x_n & x_{n+1} \\ y_n & y_{n+1} \end{vmatrix}\right) \qquad (2.2)$$

$$c_y = \frac{1}{6A}\sum_{n=0}^{N-1}\left(y_n + y_{n+1}\begin{vmatrix} x_n & x_{n+1} \\ y_n & y_{n+1} \end{vmatrix}\right) \qquad (2.3)$$

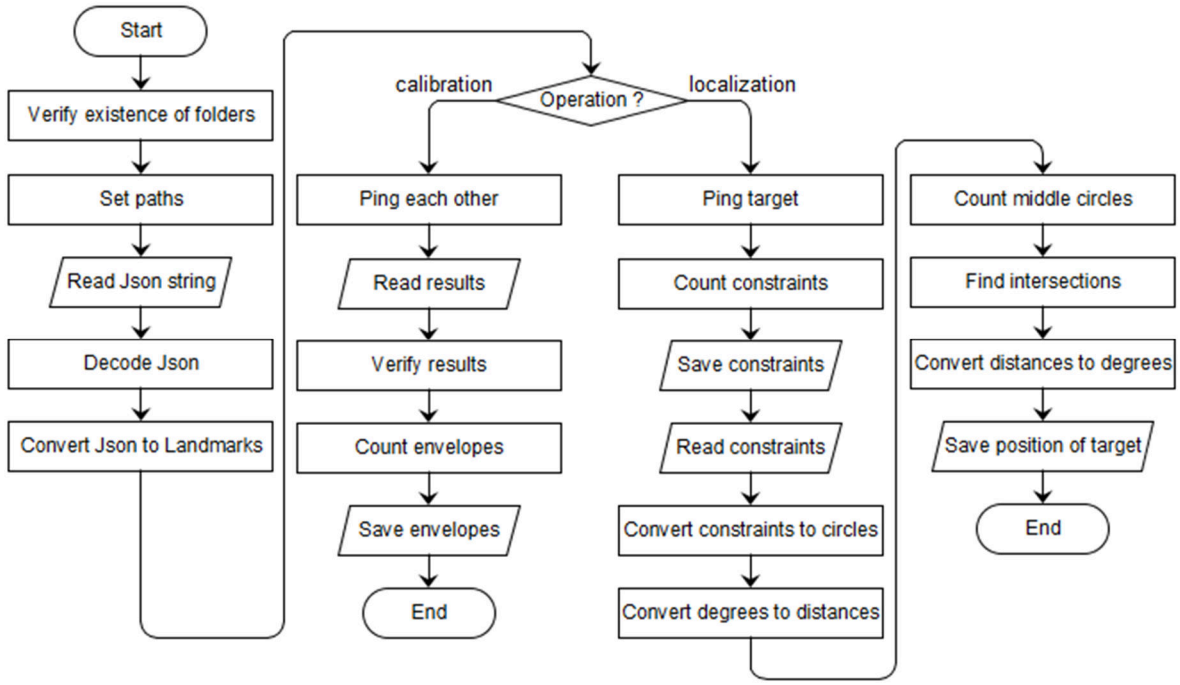
2.1.10 Třída JsonString

Tato poslední třída obsahuje tři řetězcové proměnné Ip, Lat a Lng pro uložení IP adresy a zeměpisné šířky a délky. Vstupní Json řetězec z příkazové řádky při startu celého programu je dekódován pomocí knihovny System.Web.Helpers.dll, respektive pomocí metody Decode z této knihovny a výstup tohoto dekódování je pak uložen do seznamu Json záznamů vytvořených z výše popsaných proměnných. Ve třídě JsonString je také vytvořena jedna metoda

- **jsonToLandmarks** pro převedení dekódovaného řetězce Json na seznam referenčních bodů.

2.1.11 Hlavní program

Hlavní program projektu je rozdělen na tři části. První část je společná, další část je pro kalibraci a poslední pro lokalizaci. To, zda se bude provádět kalibrace nebo lokalizace, určuje stav parametru O. Činnost všech tří částí hlavního programu je zobrazena na Obr. 2.2, která je poté popsána.



Obr. 2.2 Vývojový diagram programu Octant

Společná část hlavního programu

V této části jsou nastaveny řetězcové proměnné pro cesty k souborům, jako jsou cesta k hlavnímu logu o aktuálních stavech programu, cesta k souboru s hranicemi omezení a cesty k horním a dolním obálkám. Poté jsou ověřeny existence potřebných složek a v případě potřeby jsou tyto složky vytvořeny, popřípadě je vymazán starý soubor hlavního logu. V dalších krocích už následuje načtení a deserializace neboli dekódování řetězce Json, ve kterém jsou zadány referenční body, které budou použity pro kalibraci a lokalizaci. Dekódovaný Json je pak převeden na seznam referenčních bodů.

Část kalibrace

Tato část programu se provede, pokud je stav parametru -O z příkazové řádky nastaven na „calibration“. V této části je pomocí vláken spuštěna metoda pro zjištění vzájemného zpoždění mezi referenčními body zadanými v řetězci Json. Čeká se, dokud neskončí všechna vlákna a poté je opět pomocí vláken spuštěna metoda pro načtení výsledků, které si každý referenční bod u sebe uložil. Opět se čeká na dokončení práce všech vláken. Poté jsou zkontrolovány výsledky uložených zpoždění. Po promazání nevhodných výsledků je na řadě poslední krok a tím je výpočet obálek. Jsou spuštěny potřebné metody a obálky jsou uloženy do souboru. Tímto tato část programu končí.

Část lokalizace

Tato část programu se provede, pokud je stav parametru -O z příkazové řádky nastaven na „localization“. Pomocí vláken je spuštěna metoda pro zjištění zpoždění od cíle ke všem referenčním bodům zadanými v řetězci Json. Čeká se na skončení všech vláken a poté jsou u všech referenčních bodů spočteny hranice omezení. Tyto hranice jsou uloženy do souboru a může začít počítání průsečíků. Hranice omezení jsou ze souboru načteny do seznamu uzlů a ten je převeden do seznamu kružnic.

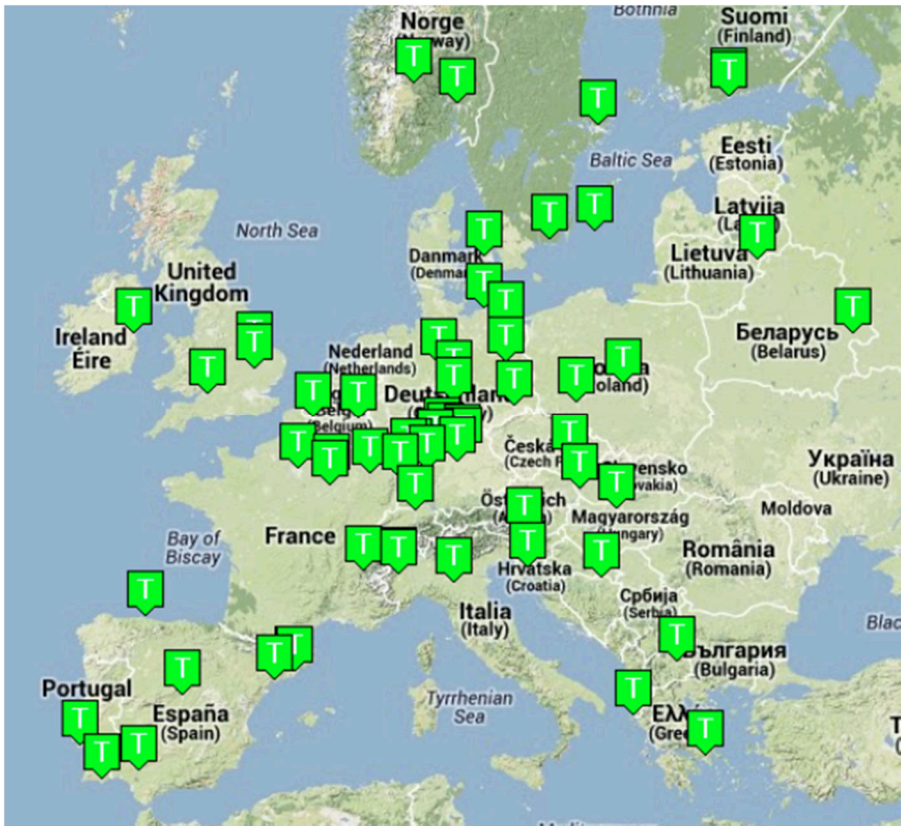
Nyní jsou vytvořeny kružnice se středy v polohách jednotlivých referenčních bodů. Stupně zeměpisných délek a šířek těchto středů jsou převedeny na vzdálenosti a jsou spočteny střední kružnice. Následně je provedeno nalezení všech potřebných průsečíků. Mohou nastat čtyři případy:

- žádný průsečík nenalezen - v tomto případě metoda nevrátí žádnou hodnotu výsledku. Pravděpodobně nebyl vypočten dostatek kružnic omezení pro určení cílové polohy, nebo se prostě kružnice neprotnuly v žádném bodě,
- nalezen jeden cílový průsečík – tento je brán jako odhadnutá poloha cíle, vzdálenosti jsou převedeny zpět na stupně a výsledek je vypsán na výstup,
- nalezeny dva cílové průsečíky - je proveden přepočet ze dvou průsečíků a poté převod vzdáleností zpět na stupně. Následně je výsledek vypsán na výstup.
- nalezeno více průsečíků – tehdy je proveden výpočet oblasti pravděpodobného výskytu cíle a zjištěno těžiště této oblasti, které je uloženo jako odhadnutá poloha hledaného cíle. Tato poloha je opět převedena zpět na stupně a výsledek je vypsán na výstup.

2.2 Testování metody Octant s reálnými daty

Před otestováním naprogramovaného algoritmu bylo nutné vybrat referenční orientační body a cíle. To bylo provedeno na internetových stránkách [22], které spravuje Ing. Lukáš Verner. Zde bylo vybráno 20 referenčních orientačních bodů na Obr. 2.3 a dostupné cíle na Obr. 2.4. Oba tyto obrázky jsou převzaty z výše zmíněného serveru [22].

Obr. 2.3 Vybrané referenční body pro lokalizaci



Obr. 2.4 Vybrané cíle pro lokalizaci

Polohy jak orientačních bodů, tak cílů jsou známy, takže bylo možné výsledky metody porovnat se skutečnou polohou hledaných cílů. Výsledky lokalizací těchto cílů pak byly porovnány i s ostatními lokalizačními metodami uloženými na výše uvedených stránkách, kde jsou uloženy i funkční programy těchto metod a mohou zde být spouštěny. Obecný příkaz pro spuštění metody v příkazové řádce je:

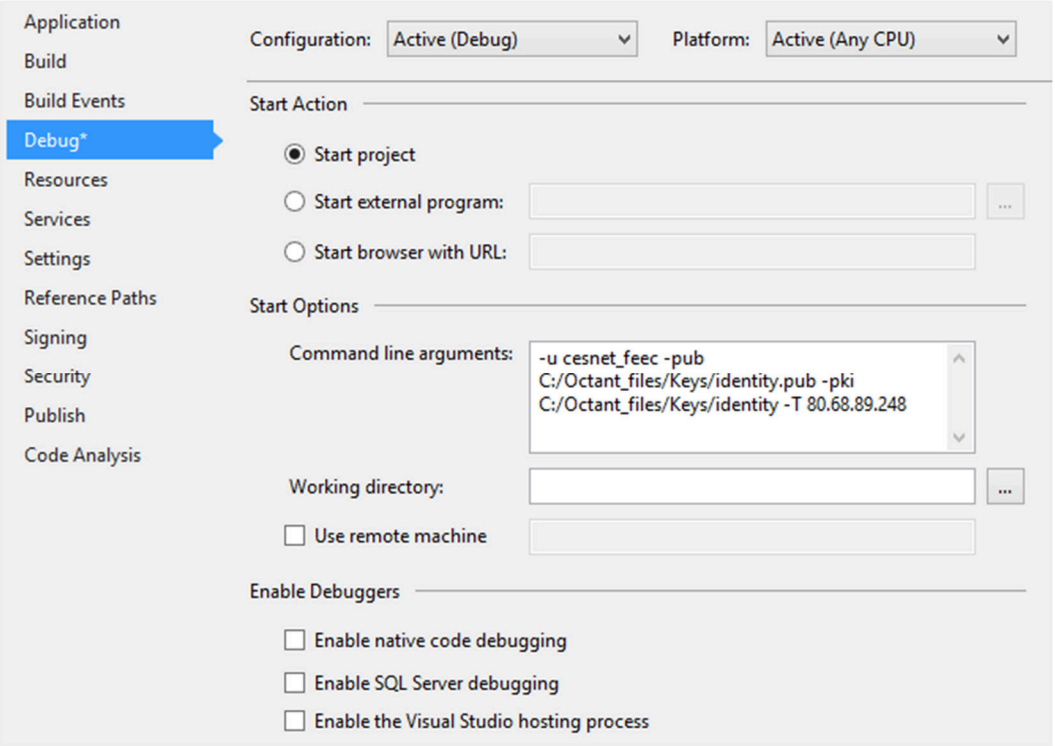
```
-u USER -pub PUB} -pki PKI -T TARGET -L JSON_LANDMARKS -O OPERATION.
```

Parametr `USER` je uživatel, který má povoleno se připojit přes SSH k referenčním bodům, parametr `PUB` je cesta k veřejnému klíči, `PKI` cesta k tajnému klíči, `TARGET` je IP adresa hledaného cíle, `JSON_LANDMARKS` je seznam referenčních bodů a parametr `OPERATION` je volba operace mezi kalibrací a lokalizací.

2.2.1 První fáze testování

První fáze testování probíhala ve vývojovém prostředí, ve kterém byl program vyvíjen, tedy ve Visual Studiu 2012 pod operačním systémem Windows 8.1. Parametry příkazové řádky

byly zadány v nastavení projektu, viz Obr. 2.5. Protože byla velikost vstupních dat pole `Command line arguments` omezena, musel být seznam referenčních bodů uložen přímo v programu. Parametr pro operaci zde taktéž uveden není, protože v této fázi testování byly obě operace prováděny hned za sebou.



Obr. 2.5 Nastavení projektu Octant v prostředí Visual Studio 2012

2.2.2 Druhá fáze testování

Po úspěšném odzkoušení metody v operačním systému Windows mohlo začít testování v linuxové distribuci CentOS (Community Enterprise Operating Systems) ve virtuálním stroji. Program byl upraven a rozdělen na dvě hlavní části a to kalibrační a lokalizační. Díky tomu už začal být používaný parametr `-O`, který rozhoduje, která část programu bude spuštěna. Ještě před tím musel být nainstalován program Mono pro spuštění .NET aplikace pod linuxovým operačním systémem [24]. Zde se vyskytl problém, kdy program pořád vypisoval chybová hlášení, že nemůže načíst knihovny. Problém však nebyl v programu, ale v nastavení vývojového prostředí, kde se muselo zatrhnout políčko `Enable the Visual Studio hosting proces` na Obr. 2.3.

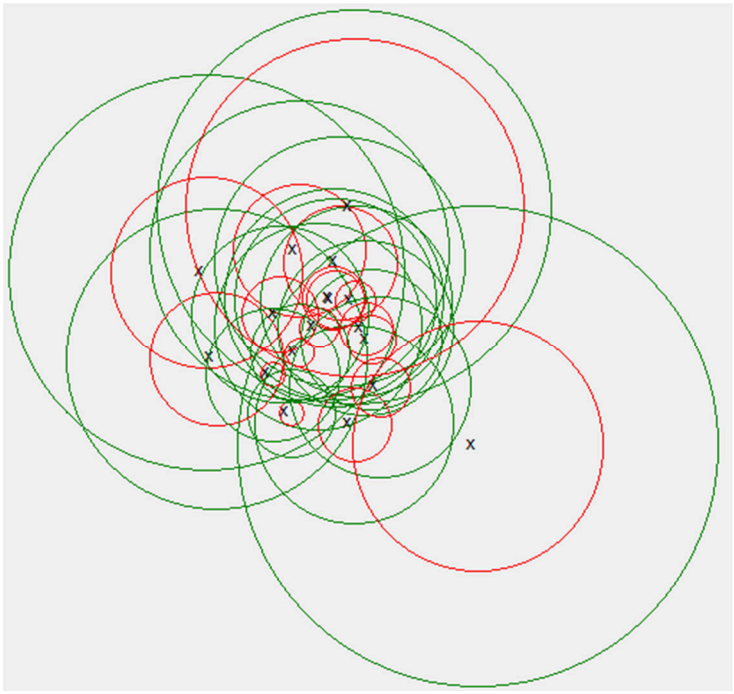
2.2.3 Ukázka lokalizace vybraného cíle

Pro ukázku lokalizace cílové stanice byla použita vybraná IP adresa cíle 195.113.0.2 s polohou [49,22684°,16,59615°]. Dále, jak již bylo zmíněno výše, bylo vybráno 20 referenčních bodů. Příkaz pro kalibraci je uveden na Obr. 2.6.

```
/usr/local/bin/mono /root/OctantServer2.exe -u 'cesnet_feec' -pub '/root/Octant/Keys/identity.pub' -pki '/root/Octant/
Keys/identity' -T '195.113.0.2' -L '[
{"ip":"130.83.166.245","name":"host2.planetlab.informatik.tu-darmstadt.de","lat":"49.53","lng":"8.4"},
{"ip":"152.66.245.161","name":"planetlab1.tmit.bme.hu","lat":"47.4725","lng":"19.6"},
{"ip":"193.63.75.21","name":"planetlab-4.imperial.ac.uk","lat":"51.1","lng":"-0.1"},
{"ip":"195.113.161.13","name":"ple1.cesnet.cz","lat":"50.102","lng":"14.3916"},
{"ip":"130.37.193.141","name":"planetlab1.cs.vu.nl","lat":"52.35","lng":"4.9"},
{"ip":"193.0.109.25","name":"prata.mimuw.edu.pl","lat":"52.211","lng":"20.981"},
{"ip":"131.254.208.12","name":"inriarennes1.irisa.fr","lat":"48.1155","lng":"-1.65071"},
{"ip":"194.254.215.11","name":"planetlab1.utt.fr","lat":"48.2693","lng":"4.06739"},
{"ip":"160.80.221.37","name":"planet-lab-node1.netgroup.uniroma2.it","lat":"41.8556","lng":"12.43"},
{"ip":"212.201.44.81","name":"planetlab1.eecs.jacobs-university.de","lat":"53.04","lng":"8.49"},
{"ip":"130.104.72.201","name":"onelab2.info.ucl.ac.be","lat":"50.6833","lng":"4.61667"},
{"ip":"138.246.99.250","name":"planetlab2.lkn.ei.tum.de","lat":"48.1493","lng":"11.69"},
{"ip":"193.144.21.131","name":"planetlab2.urv.cat","lat":"41.07","lng":"1.15"},
{"ip":"192.42.43.23","name":"planetlab2.unineuchatel.ch","lat":"46.59","lng":"6.56"},
{"ip":"193.190.168.51","name":"planetlab2.extern.kuleuven.be","lat":"50.8625","lng":"4.68599"},
{"ip":"193.138.2.13","name":"planetlab-13.e5.ijs.si","lat":"46.0423","lng":"14.488"},
{"ip":"193.1.201.27","name":"planetlabeu-2.tssg.org","lat":"52.2441","lng":"-7.15825"},
{"ip":"62.108.171.76","name":"ple2.tu.koszalin.pl","lat":"54.2047","lng":"16.1972"},
{"ip":"193.166.167.5","name":"planetlab2.rd.tut.fi","lat":"61.449","lng":"23.855"},
{"ip":"141.22.213.34","name":"merkur.planetlab.haw-hamburg.de","lat":"53.5686","lng":"10.0386"}]' -O calibration
```

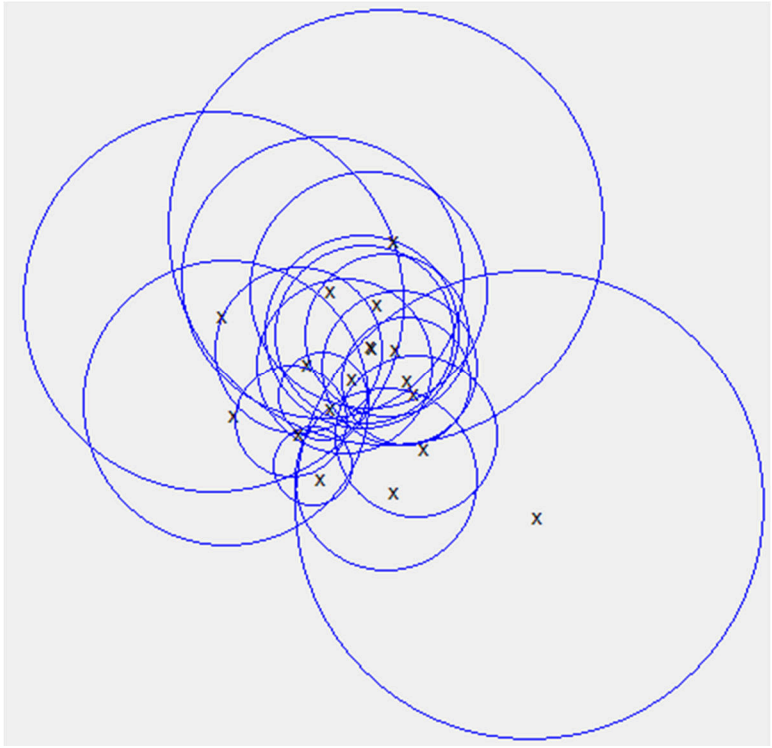
Obr. 2.6 Příkaz pro spuštění programu

Po kalibraci byl spuštěn příkaz pro lokalizaci, kde se jen změnil v příkazu parametr `-O` na `localization`. Grafická reprezentace negativních a pozitivních kružnic pro tento případ je zobrazena na Obr. 2.7. Červeně jsou znázorněny negativní kružnice, zeleně pak pozitivní.

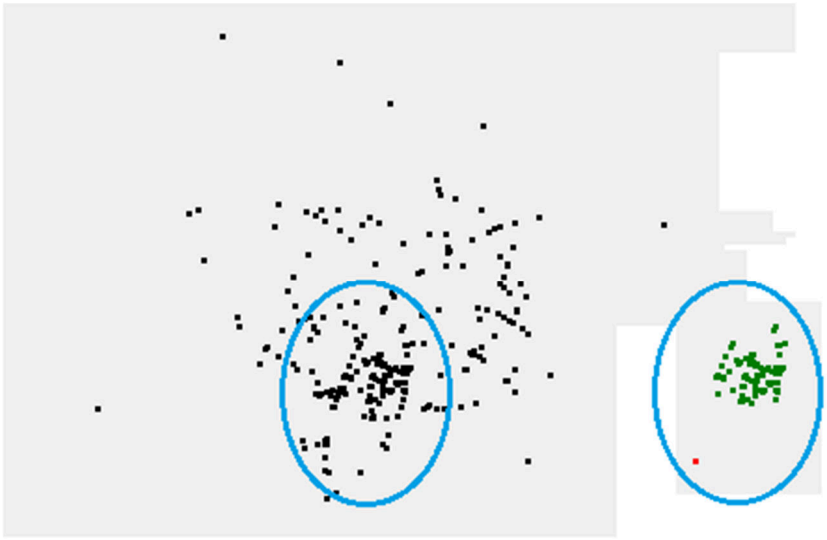


Obr. 2.7 Hranice omezení pro zvolených 20 referenčních bodů

Zde je však zobrazených uzlů jen 18, protože ke zbývajícím dvěma uzlům se v daném testu nebylo možno připojit přes SSH, nebo nebylo možné vypočítat hranice omezení pro dané uzly. Přepočtené střední kružnice jsou na Obr. 2.8. Na dalším Obr. 2.9 jsou pak zobrazeny v jeho levé části všechny nalezené průsečíky kružnic a v jeho pravé části pak průsečíky ležící ve všech kružnicích, které jsou vyznačeny zeleně. Modrými elipsami je v tomto obrázku vyznačena důležitá oblast v obou částech. Červeně je označen vypočtený cílový průsečík.

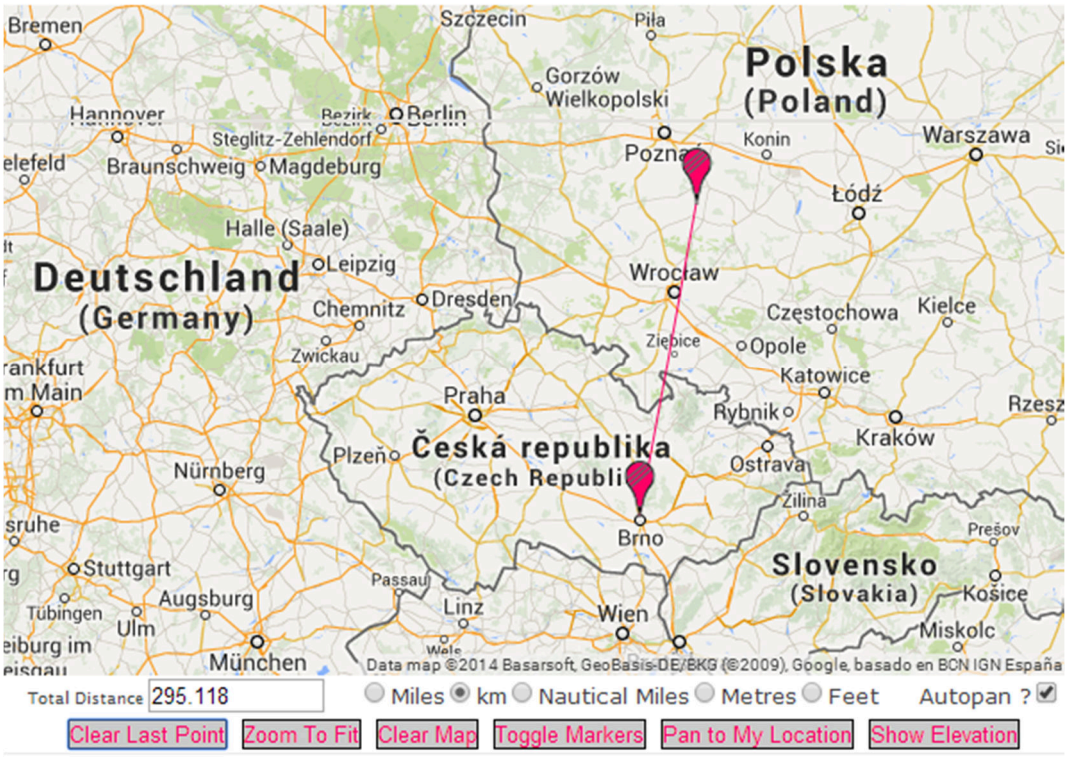


Obr. 2.8 Střední kružnice pro zvolených 20 referenčních bodů



Obr. 2.9 Průsečíky všech středních kružnic

Výsledek lokalizace byl pro hledaný cíl zobrazen na výstupu ve formátu Json jako `{"lat":"51.83571","lng":"17.33871"}`. Vzdálenost mezi skutečnou a lokalizovanou polohou byla 295.118km, viz Obr. 2.10 (převzato z [23]).



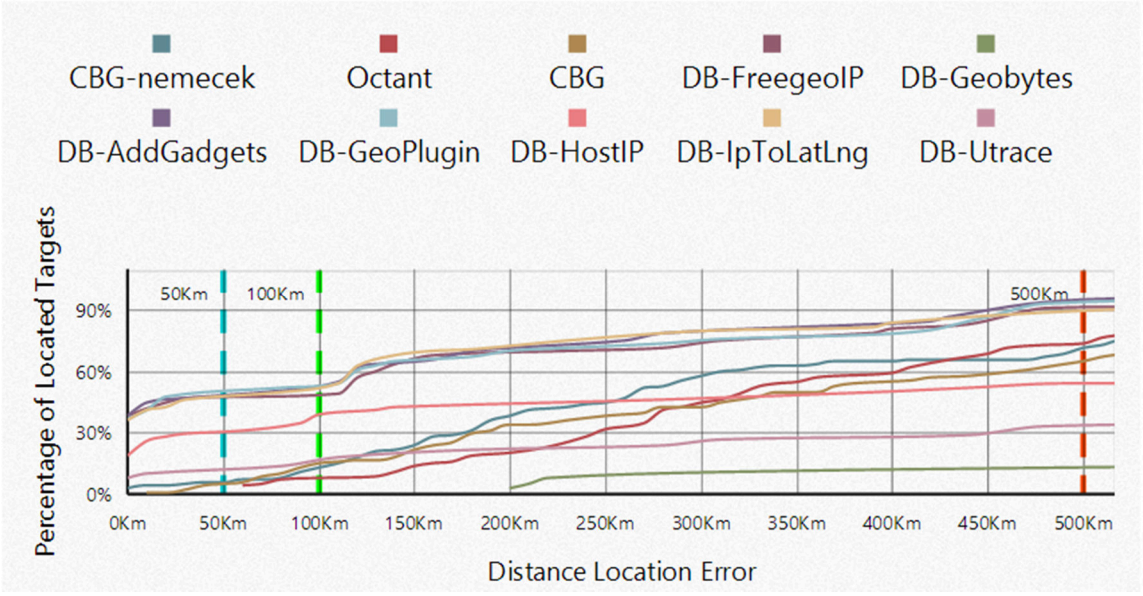
Obr. 2.10 Vzdálenost mezi skutečnou a lokalizovanou polohou

2.3 Srovnání metody Octant s ostatními metodami

Výsledky metody Octant byly porovnány s výsledky dalších metod dostupných na stránkách [22]. Je to aktivní geolokační metoda CBG (David Ripper a bc. Ladislav Němeček) a pasivní IP geolokace (Andrea Mrázová). Na těchto stránkách byly také pomocí užitečného nástroje pro tvorbu grafů zhotoveny dvě závislosti na základě těchto hodnot. Jsou jimi kumulativní distribuční funkce (Cumulation Distribution Function - CDF) a statistika chyb lokalizace pro všechny metody.

2.3.1 Kumulační distribuční funkce

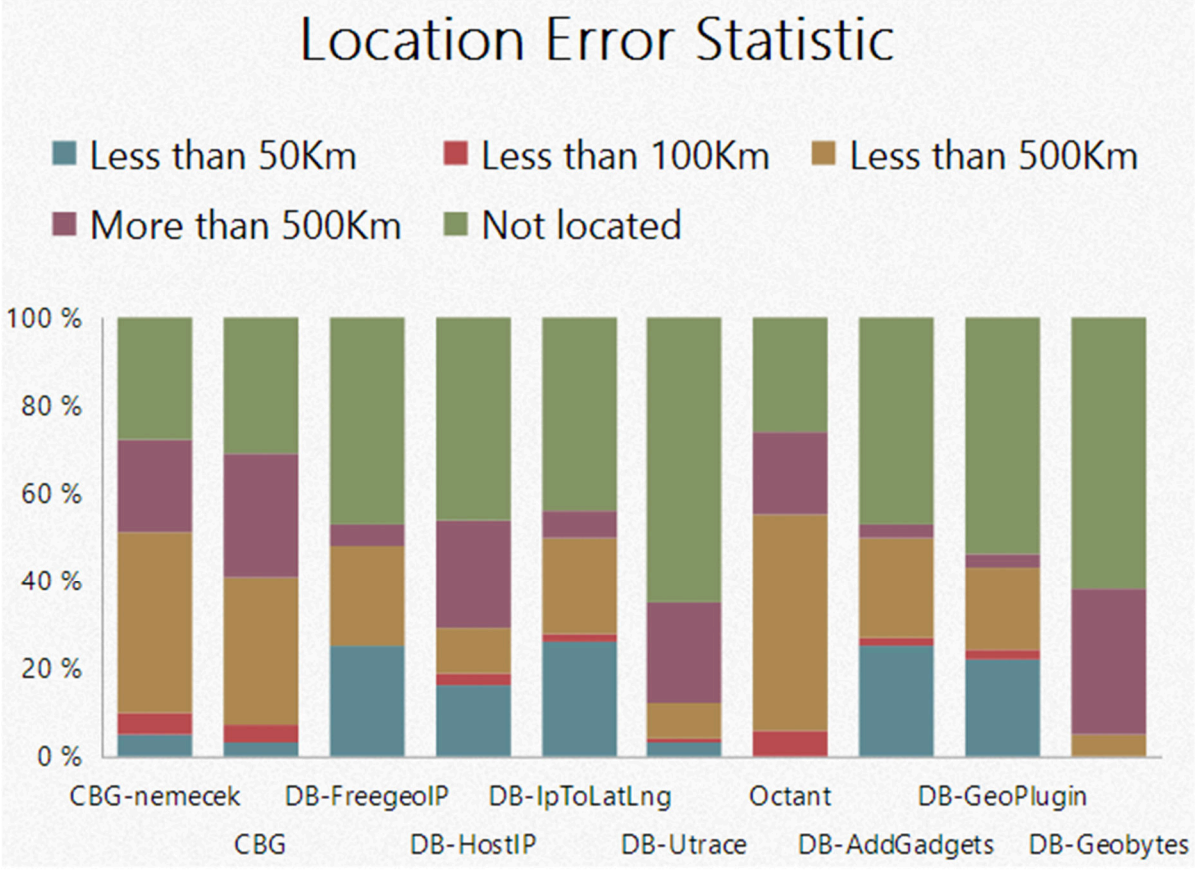
Kumulativní distribuční funkce vyjadřuje závislost procentuální pravděpodobnosti výskytu cíle na průměrné chybě přesnosti lokalizace. Počítá zde pouze s případy, kdy metoda dokáže cíl lokalizovat. Hodnota 500km už je považována za chybu lokalizace. Z Obr. 2.11 je patrné, že metoda Octant je v nízkých hodnotách vzdáleností druhá nejhorší, u hranice 500km se pak její procentuální pravděpodobnost výskytu cíle zvýšila. Pasivní geolokační metody s využitím databáze IpToLatLng, AddGadgets, GeoPlugin a FreeGeoIP jsou po celou dobu do hranice 500km lepší než ostatní metody. Aktivní metody sice nedosahují přesnosti pasivních metod, mají ale výhodu v tom, že mohou lokalizovat jakýkoliv počítač, který dokáže odpovědět na ICMP dotaz.



Obr. 2.11 Kumulativní distribuční funkce všech měřených metod

2.3.2 Statistika chyb lokalizace

Tento graf udává, kolik procent uzlů bylo lokalizováno v jaké chybové vzdálenosti. Chyba menší než 50km značí přesnost na úrovni města, menší než 100km na úrovni regionu, menší než 500km na úrovni státu a větší než 500 km je považováno za chybu lokalizace. Z Obr. 2.12 lze jednoduše vyčíst, že metoda Utrace nebyla schopna najít polohu nejvíce hledaných cílů. Metoda Octant naopak v tomto směru byla na prvním místě. A také lokalizovala nejvíce cílů s chybou menší než 500km. Nejvíce výsledků s chybou větší než 500 km vyprodukovala metoda Geobytes a metoda CBG. U metody Octant mohlo prázdný výsledek zapříčinit špatně změřené zpoždění od cíle a tím pádem i špatné hranice omezení, což mohlo způsobit neprotnutí kružnic a nenalezení odpovídající oblasti s největší pravděpodobností výskytu.



Obr. 2.12 Srovnání chybovosti s ostatními metodami

Závěr

Všechny cíle, které byly popsány v úvodu této práce, se podařilo splnit. V teoretické části této diplomové práce je nejprve představen základní pojem geolokace a její účel. Následně jsou popsána nejdůležitější fakta a principy týkající se základních pasivních a aktivních geolokačních metod. Nejvíce pozornosti zde bylo věnováno hlavnímu tématu této diplomové práce – aktivní lokalizační metodě Octant, která oproti základním metodám využívá řadu pokročilejších technik (využití negativních hranic referenčních uzlů apod). Dále byla v teoretické části představena experimentální síť PlanetLab a poskytnuty jen nejzákladnější informace o programovacím jazyku C#.

V praktické části práce byla podrobně popsána struktura programu realizující metodu Octant. Nejdříve byly popsány navrhnuté třídy, které byly využity a poté jejich nejdůležitější metody a proměnné. Popsán byl zde i nejdůležitější příkaz pro zjišťování zpoždění od ostatních uzlů. Dále byl v této části popsán vývojový diagram celého programu po částech a jeho možné výsledky při lokalizaci cílové stanice.

Dále bylo v praktické části popsáno testování na reálných datech. Testování probíhalo v několika fázích. V té první se testovala funkčnost algoritmu ve vývojovém prostředí Visual Studio pod operačním systémem Windows, kde byl prográm také vyvíjen. V druhé fázi testování se pak přešlo pod linuxový operační systém CentOS, kde byl program testován už z příkazové řádky. Po správném nastavení kompilátoru Visual Studia pak bylo testování lokalizace prováděno zde.

V poslední části praktické části je metoda podrobena srovnání s ostatními aktivními a pasivními lokalizačními metodami. Bylo určeno 20 referenčních orientačních bodů a vybrány všechny aktivní cíle z [18]. Výsledky přesnosti metody Octant byla srovnána na základě průměrné chyby přesnosti (tj, vydálenost odhadnuté polohy cíle od jeho skutečné polohy), kde metoda značně zaostávala za pasivními metodami a na základě procentuálního vyjádření počtu lokalizovaných uzlů s danou chybou v kilometrech. Metoda Octant nebyla schopna lokalizovat stanice na úrovni města ve srovnání s CBG metodami, kde byl jejich úspěch zhruba u 5% hledaných stanic. U pasivních metod byla úspěšnost na úrovni města mnohem vyšší a někde dokonce přesahovala 20%. Z obou srovnání je patrné, že většina pasivních metod je přesnější než aktivní metody, což odpovídá poznatkům popsaných v teoretické části. Pasivní metody však nedokázaly lokalizovat velký počet cílů, což poukazuje na hlavní

nevýhody pasivních metod. Databáze zde musí být manuálně spravovány, data v nich obsahují chyby a často se musí aktualizovat.

Seznam použitých zdrojů

[1] VERNER, L., KOMOSNÝ, D. *Geolokace síťových zařízení v internetových sítích*. Elektrorevue [online]. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2011. 7 stran. [Citace: 2014-05-02] Dostupný z WWW: http://www.elektrorevue.cz/cz/download/geolokace-sitovych-zarizeni-v-internetovych-sitich/

[2] ČELEDA, P., KADERKA, J. *Geolokace a bezpečnost počítačových sítí*. [online]. Praha: TATE International, s.r.o., 2012. 8 stran. [Citace: 2014-05-02] Dostupný z WWW: https://is.muni.cz/repo/1067555/geolokace-a-bezpecnost-pocitacovych-siti.pdf?lang=cs

[3] BALEJ, J. *Srovnání přesnosti aktivních geolokačních technik*. [online]. Brno: Mendelova univerzita v Brně, Provozně ekonomická fakulta, Ústav informatiky, 2012. [Citace: 2014-05-02] Dostupný z WWW: http://access.fel.cvut.cz/view.php?nazevclanku=srovnani-presnosti-aktivnich-geolokacnich-technik&cisloclanku=2012070001

[4] WHOIS-SEARCH.COM *The domain name experts.* [online]. 2014. Dostupný z WWW: http://www. whois-search.com/

[5] GEO IP TOOL [online]. Dostupný z WWW: http://www.geoiptool.com/

[6] NETWORK TOOLS.COM [online]. Dostupný z WWW: http://www.network-tools.com/

[7] NMONITORING *DNS dig nslookup Online.* [online]. Dostupný z WWW: http://dig-nslookup.nmonitoring.com/

[8] PING.EU [online]. Dostupný z WWW: http://www.ping.eu/

[9] KATZ-BASSETT, E., JOHN, J. P., KRISHNAMURTHY, A., WETHERALL, D., ANDERSON, T., CHAWATHE, Y. *Towards IP Geolocation using Delay and Topology Measurements*. [online]. Rio de Janeiro, 2006. 13 stran. [Citace: 2014-05-02] Dostupný z WWW: http://djw.cs.washington.edu/papers/imc2006-geoloc.pdf

[10] ARIF, M. J., KARUNASEKERA, S., KULKARNI, S. *GeoWeight: Internet Host Geolocation Based on a Probability Model for Latency Measurements*. [online]. Melbourne: University of Melbourne, Department of Computer Science and Software Engineering, 2010. 10 stran. [Citace: 2014-05-02] Dostupný z WWW: http://crpit.com/confpapers/CRPITV102Arif.pdf

[11] LAKI, S., MÁTRAI, P., HÁGA, P., SEBOK, T., CSABAI, I., VATTAY, G. *Spotter: A model based active geolocation service.* [online]. Budapest, Hungary: Eötvös Loránd University, 2011. 9 stran. [Citace: 2014-05-02] Dostupný z WWW: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5935165

[12] WONG, B., STOYANOV, I., SIRER, E. G. *Octant: A Comprehensive Framework for the Geolocalization of Internet Hosts.* [online]. Ithaca, NY: *Dept. of Computer Science,*

Cornell University, 2007. 14 stran. [Citace: 2014-05-02] Dostupný z WWW: http://www.cs.cornell.edu/people/egs/papers/octant-nsdi.pdf

[13] VENESS, CH., *Calculate distance, bearing and more between Latitude/Longitude points.* [online]. 2010. [Citace: 2014-05-02] Dostupný z WWW: http://www.movable-type.co.uk/scripts/latlong.html.

[14] VOJÁČEK, CH., *Analytická geometrie - Směrnicová rovnice přímky.* [online]. 2008. [Citace: 2014-05-02] Dostupný z WWW: http://maths.cz/clanky/analyticka-geometrie-smernicova-rovnice-primky.html

[15] NAVRÁTIL, J., *PlanetLab – model budoucího internetu .* [online]. 2006. [Citace: 2014-05-02] Dostupný z WWW: http://www.ics.muni.cz/bulletin/articles/525.html

[16] ONELAB, *PlanetLab European Tutorial.* [online]. 2008. 114 stran. [Citace: 2014-05-02] Dostupný z WWW: http://www.ict-fireworks.eu/fileadmin/events/FIREweek/ PL_1. pdf

[17] PLANETLAB, *An open platform for developing, deploying, and accessing planetary-scale services – User's Guide.* [online]. 2002. [Citace: 2014-05-02] Dostupný z WWW: http://www.planet-lab.org/doc/guides/user

[18] PLANETLAB, *An open platform for developing, deploying, and accessing planetary-scale services.* [online]. 2002. [Citace: 2014-05-02] Dostupný z WWW: http://svn. planet-lab.org/

[19] BĚHÁLEK, M., *Programovací jazyk C#.* [online]. Ostrava: Vysoká škola báňská Ostrava, Katedra informatiky. 2007. [Citace: 2014-05-02] Dostupný z WWW: http:// www.cs.vsb.cz/behalek/vyuka/pcsharp/text/pr01.html

[20] DVOŘÁK, F., *Využití znalosti topologie páteřních sítí pro určování fyzické polohy stanic v síti Internet.* [online]. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2012. [Citace: 2014-05-02] Dostupný z WWW: https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=52206, Diplomová práce.

[21] GUEYE, B., ZIVIANI, A., CROVELLA, M., *Constraint-Based Geolocation of Internet Hosts.* [online]. 2007. 14 stran. [Citace: 2014-05-02] Dostupný z WWW: http://dl.acm.org/citation.cfm?id=1217693

[22] VERNER, L., *IP Geolocation - Active IP Geolocation Methods Development Tool.* Dostupný z WWW: http://betka.utko.feec.vutbr.cz/

[23] DAFTLOGIC, *Distance calculator.* Dostupný z WWW: http://www.daftlogic.com /projects-google-maps-distance-calculator.htm

[24] *Installing Mono in CentOS 5.x/6.x.* 2013. Dostupný z WWW: http://wiki.phonicuk. com/Installing-Mono-in-CentOS-5-x.ashx

Seznam použitých zkratek

CBG	Constraint Based Geolocation
CDF	Cumulation Distribution Function
CentOS	Community Enterprise Operating Systems
DNS	Domain Name Systém
GSM	Global Systém for Mobile
IANA	Internet Assigned Numbers Authority
ICMP	Internet Control Message Protocol
IP	Internet Protocol
MAC	Media Access Control
OSI	Open Systems Interconnection
PI	Principal Investigator
RTT	Round-Trip Time
SOI	Speed of Internet
SSH	Secure Shell
SSID	Service Set Identifier
TBG	Topology Based Geolocation
VoIP	Voice over Internet Protocol

Obsah CD

Přiložený disk obsahuje následující položky:

- Elektronickou verzi diplomové práce.
- Soubor Vysledky.xlsx se všemi výsledky pro testované uzly.
- Program OctantServer2 a jeho zdrojový kód.